Izdaje BIGZ OOUR „Duga“

računari

oktobar
1985.

Specijalno izdanje časopisa „Galaksija“

8

Cena 250 dinara
Izlazi jednom mesečno

Računari u izlogu **ATARI 520 ST**

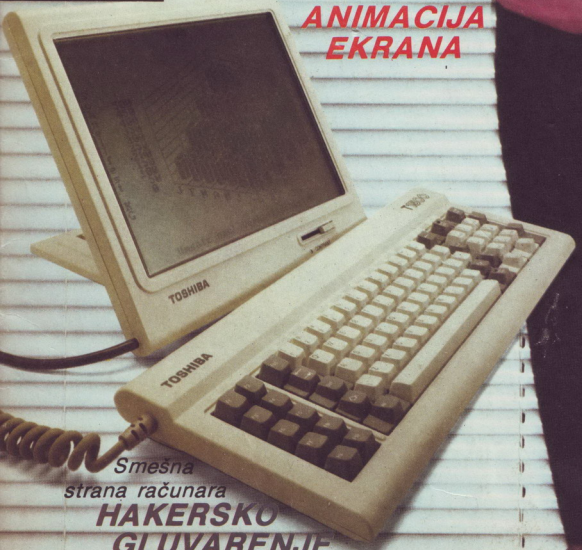
Periferijska oprema **MODEMI**

Ekskluzivno „Spektrum“ **EKRANSKI EDITOR**

Umetak na 32 strane **SVE „KOMODOROVE“**

RUTINE

„amstrad“
ANIMACIJA EKRANA



Smešna strana računara
HAKERSKO GLUVARENJE

8

računari

Izlazi jednom mesečno u izdaje BIGZ OOUR „Duga"

Specijalno izdanje časopisa „Galaksija"

Cena 250 dinara / oktobar 1985.

Izdaje
Beogradski izdavačko-grafički zavod
OOUR Novinska delatnost „Duga"
11000 Beograd
Bulevar vojvode Mišića 17

Telefoni
650-161 (redakcija)
650-328 (prodaja)
651-793 (propaganda)

Generalni direktor
Dobrosav Petrović

Direktor OOUR „Duga"
Bratoljub Babić
Glavni i odgovorni urednik
Gavrilo Vučković
Urednik izdanja
Jova Regašek

Likovna i grafička oprema:
Dušan Milatović

Redakcija časopisa „Galaksija"
Tanasije Gavranović, pomoćnik glavnog i odgovornog urednika
Esad Jakupović, zamenik glavnog i odgovornog urednika
Aleksandar Milinković, urednik
Jova Regašek, urednik
Zorka Simović, sekretar redakcije
Sydan Stojančev, novinar
Gavrilo Vučković, glavni i odgovorni urednik

Stručna saradnja
Dejan Ristanović
Nevenka Spalević
Anđelko Zgonjanin
Mihajlo Tešević

Autori tekstova:

Aleksandar Demić
Branko Đaković
Mihajlo Karapandžić
Vladimir Kostić
Vladimir Krstonošić
Vladimir A. Mihajlović
Ivan Nador
Bogdan Petrović
Petar Putnik
Dejan Ristanović
Jelena Rupnik
Jovan Skuljan
Zoran Životić
Živanimir Vistrička

Crtezi
Miša Marković

Tehnička saradnja
Ljubiša Milovanović
Lev Rjadčenko

Prevodioci
Esad Jakupović
Ksenija Pješčić-Lebedinski
Domagoj Bačić

Izdavački savet „Galaksije"

Dr Rudi Debijadi, prof. dr Branislav Dimitrijević (predsednik), Radovan Draškovic, Tanasije Gavranović, Živorad Glišić, Esad Jakupović, Velizar Mašlać, Nikola Pajić, Željka Perunović, prof. dr Momčilo Ristić, Vlada Ristić, dr inž Milorad Teofilović, Vidojko Veličkovic, Velimir Vesović, Miloje Vuković

Štampa
Beogradsko izdavačko-grafički zavod
11000 Beograd, Bulevar vojvode Mišića 17

Žiro-račun kod SDK 60802-833-2463
Devizni račun kod Beobanke
60811-620-6-92701-999-01066
Za inostranstvo cena dvostruka (400 D, 2,50 US$, 6,50 DM, 45 Sch, 5,50 Sfrs, 20 FFRS)

Na osnovu mišljenja Republičkog sekretarijata za kulturu broj 413-77-72-03 i „Službenog glasnika broj 26-72, ovo izdanje oslobođeno je poreza na promet

Sadržaj

3/ šta ima novo

6/ **naš test**
„galaksija plus"

9/ **računari u izlogu**
„atari 520 ST"

12/ **računari u razgovoru**
ima li programera u avionu?

14/ **smešna strana računara**
hakersko gluvarenje

17/ **periferijska oprema**
svet na dlanu

20/ **istorija računara**
radanje programskih jezika

23/ **akcije**
ekranski editor i druge bajke

43/ **biblioteka programa**

47/ **računari i matematika**
grafičko predstavljanje funkcija

51/ **računari iz mog ugla**
kako preživeti računare

52/ **programiranje u bejziku**
na vrh brda vrba mrda

56/ **majstorije na računaru**
mašinska veza

58/ **radionica logičnih igara**
volite li paskal

60/ **računari u poslovnoj primeni**
obrada ličnih dohodaka

63/ **hakeri u nevolji**

64/ **računari i umetnost**
spektroteka na kompjuteru

Jedanput mesečno

Navršava se, ovih dana, dve godine kako smo pokrenuli „Računare" i lansirali računar „galaksija". Za nas, koji smo radili na tim poslovima, to vreme je prošlo nestvarno brzo — za one koji su nas čitali nestvarno sporo, iako znamo da su ljubitelji računara, po prirodi, veoma nestrpljiv svet i da svojim ritmom izlaženja u mene pa na ušteo ne možemo da pratimo njihove impulse, oduvek nam je bilo važnije šta kazujemo nego kada to činimo. Potpuno spremljeni „Računari" znali su ne jednom da čekaju da neki saradnik napiše program ili tekst u koji svi verujemo.

Sve na ovom svetu, međutim, ima svoju cenu, pa i naša temeljitost. Iako smo, zahvaljujući našem stručnom saradniku i dopisniku iz Londona Andelku Zgorelcu, većinu stvari saznavali i pre nego što ih objavi svetska štampa, dešavalo se da o tome pišemo — poslednji. Period između dva izdanja „Računara" obično je dovoljan da se najavi i lansira jeden kompjuter, pobere gomila lovorovih venaca i propadne onda kada njegovim proizvodačima, naizgled, najbolje ide. O nekim stvarima nismo, čak, ni stizali da pišemo — toliko su brzo hujale kroz naš računarski život.

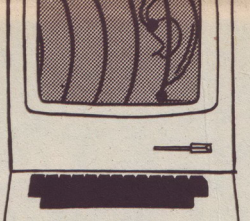
Tih programa, džojstika, računara za jednokratnu upotrebu i ostalih tema tipa „danas i nikad više" nam, istini za volju, i nije previše žao, iako bismo više voleli da smo mogli da zadovoljimo i tu sasvim prirodnu potrebu svojih čitalaca da budu u toku. Dosadašnji ritam izlaženja, međutim, vezivao nam je ruke u nekim mnogo važnijim stvarima — obavezivao nas je da budemo studiozniji i ozbiljniji nego što je to potrebno i otežavao nam da uspostavimo tešnju emotivnu vezu sa svojim čitaocima, koja je neophodna za život svakog časopisa.

Mada već poodavno prižeiljujemo da učestamo ritam „Računara", imamo utisak da smo tek sada — na radost svih onih koji navijaju da „Računari" izlaze bar jedanput dnevno — za to potpuno spremni. Ovih dana smo, zajedno sa svojim čitaocima, napravili valjda i poslednju od 6.000 „galaksija" i pritisak na redakciju „jao, pomagaj, neće da radi" nije više tako velik kao nekad. Uz Dejana Ristovića, koji je potpuno sam napisao „Računare 1" i čiji tekstovi i danas predstvljaju okosnicu svakog broja „Računara", stasala je u međuvremenu čitava ekipa mladih i darovitih autora i programera koji su čak i danas, u vreme kada svi rade samo na sporednom delu da za čitaoce „Računara" pišu ekskluzivne programe i ekskluzivne tekstove. Oko naše koncepcije okupljaju se stalno novi autori. Od ovog broja nam se pridružuje i dr Radomir A. Mihajlović, profesor programiranja u Nijagari, koji će uz stručne tekstove, saskoro početi da se javlja i sa novostima s one strane „velike bare". Time „Računari" uz Andelka Zgorelca, dobijaju još jednog stručnog dopisnika iz belog sveta, ovoga puta sa najznačajnine kompjuterske scene u svetu.

Kada smo pokretali „Računare 1", „komodor 64" je još uvek bio u dalekoj Americi, o „spektrumu" je mogio samo da se sanja, a ZX 81 je bio najpopularnija mašina na svetu. Primitivni poput tricikla, ovi računari su radili na mišiće i vijuge svojih vlasnika — tražili su od njih da ih najintenzivnije primenu puno znanja. Pojavom „mekintoša" koncept kućnog računara se iz temelja menja — od sprava za programiranje kompjuteri polako evoluiraju u sprave za korišćenje. „Atari 520 ST" i „amiga" predstavljaju prve vesnike ovog novog soja. „Amstrad" je, čak, izbacio najspecijalniji kućni računar na svetu — računar o kome njegov vlasnik ne mora ama baš ništa da zna i koji nema čak ni najobičniji bejzik — računar koji služi samo za pisanje!

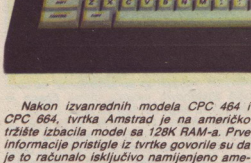
Što najbolji, čitaoci mogu da očekuju od jednomesečnih „Računara"? Da prate upravo ovakav koncept. To, pre svega, znači blagi zaokret iz voda igranja i programiranja kao intelektualnog sporta u vode programiranja da bi se računar stvarno koristio i — korišćenja računara. Uz visok standard u izboru i obradi tema — koji i predstavlja zaštitni znak — Računara" — nastojaćemo da proširimo tematsku lepezu i, istovremeno, olakšamo njihovu prezentaciju. Reču, da „Računare" još više približimo onim novinama kakve oduvek želite da čitate a niste u stanju sami da ih pravite.

Redakcija



šta ima novo

Amstrad CPC 6128 u Evropi



Nakon izvanrednih modela CPC 464 i CPC 664, tvrtka Amstrad je na američko tržište izbacila model sa 128K RAM-a. Prve informacije pristigle iz tvrtke govorile su da je to računalo isključivo namijenjeno američkom tržištu. Međutim, računalo se već pojavilo i u Evropi. Tako svega nakon malo više od godine dana od pojave prvog modela, koji je potukao Sinclairovog QL-a, Amstrad ponovo tuče konkurenciju. Što više i vlastiti model CPC 664 (449 funti) će ispasti iz trke, jer je cijena CPC 6128 (399 funti) za 50 funti niža.

Amstrad 6128 je tako u prednosti pred ostalim konkurentima i cijenom i kvalitetom: znatno je kvalitetniji od „spektruma", boljeg bejzika od komodora 64, imun na bagove koje imaju QL i Oric, te sa mnogo više memorije od BBC-a B.

Amstrad 6128 i dalje koristi Z 80 A procesor (4 MHz) tako da direktno može adresirati samo 64 K memorije, dok se ostatku pristupa pejždovanjem. Ugrađena disk jedinica koristi proširenu verziju CP/M 2.2 operativnog sistema nazvanog CP/M Plus. Što se izgleda tastature tiče — malo je promijenjen „Amstradov" koncept, ali samo na izgled. Funkcijski i kursorski tasteri postoje još uvijek, samo su gusto zbijeni uz ostali dio tastature. Zbog veličine svoje memorije, model 6128 predstavlja u malome pravi poslovni sistem. Priključći na stražnjoj strani računala najzad su dobili propisna utikačka gnezda. Tu je, naravno, i priključak za drugu disk-jedinicu koja može biti za veličinu diska od 3,35 ili 5,25 inča. Uz CP/M operativni sistem dobija se i programski jezik dr logo, te program za rukovanje grafikom — Graphic System Extension (GSX). GSX je 8-bitna varijanta Digital Research-ovog GEM grafičkog paketa. CP/M Plus je tako izveden da podržava preklapanje memorijskih stranica 64K/128K. Kod modela CPC 664, 39 K memorije koje ostaje slobodno i nije pretjerano puno čak što više ponekad je nedovoljno za ozbiljne CP/M programe kao npr. Wordstar. CP/M Plus može na CPC 6128 koristiti 61K, što je više nego dovoljno za mnoge 8-bitne vrhunske CP/M programe.

Sam po sebi zaslužima 21K memorije. Ima mnoge prednosti od ranijih varijanti — ne samo što je brži pri snimanju i učitavanju programa već sadrži neke nove instrukcije pa i preklapanje britanskog i njemačkog seta znakova. Softver sa ranijih modela se može bez problema koristiti i na ovom modelu, dok i obratno postoji velika kompatibilnost.

U ovom momentu konkurenciju „amstradu 6128" predstavlja jedino „Komodor 128", no ovaj je konkurent dosta skupliji. „Komodor 128" i „amstrad CPC 6128" predstavljaju vrhunac 8-bitnih računala.

Zvonimir Vistrička, dipl. ing.

Novi dodaci za „Amstrad"

Tvrtka Amstrad je najavila sintetizator govora (speech synthesizer) model SSA-1 koji se priključuje preko porta za disk kod računala CPC 464 ili na expansion port kod CPC 664. Naprava za svoj rad koristi programsku podršku koja se isporučuje na kazeti. Priključenjem SSA-1 ne blokira se disk port, već se na uređaj mogu priključiti druge naprave. Cijena će biti oko 30 funti.

Druga novina je asembler tvrtke Arnor Maxam ugrađen u posebnu ROM memoriju. Priključuje se na disk port ili expansion port ovisno o modelu 464 ili 664. Program u ROM-u se koristi jednostavno, a posebno je pogodno što je disk-kompatibilan. Cijena naprave je 60 funti.

Treća novost je modem Prótek 1200 pomoću kojeg se „amstrad" može povezati sa svijetom. Modem može raditi u dva režima rada. Prvi je 1200/75 Bauda, u kojem računalo prima informacije sa 1200 Bauda, a odašilje sa 75 Bauda. To je dobro za ulaz u baze podataka, ali sporo za komunikaciju korisnika s drugim korisnicima. Upravo zato na raspolaganju stoji i mogućnost rada s 1200/1200 Bauda. To je ujedno korisno za tehničke vlasnike, jer mogu komunicirati s Prestelom. Serijski međuslojp je malih dimenzija, što govori da je konstruiran s malim brojem komponenti. Pošto Prestel koristi različit format „screena", potreban je odgovarajući program za prilagodbu. Sam modem je akustičnog tipa te odgovara uz gotovo sve telefonske aparate. Napaja se baterijski, što ima za posljedicu pojavu šuma kada se baterije istroše malo ispod maksimuma.

Cijena modema, medusklopa (interfacea) i programa za prilagodbu na Prestel je oko 80 funti.

Zvonimir Vistrička, dipl. ing.

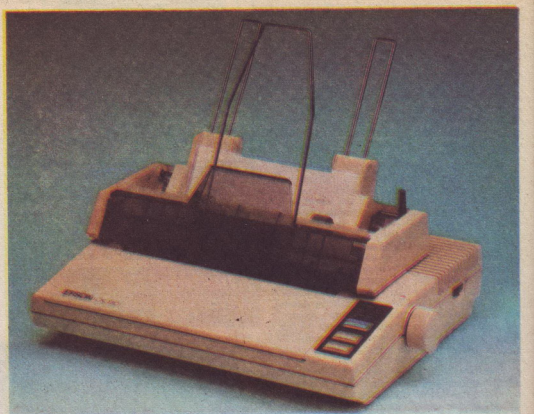
Nikad ne veruj Sinkleru

Septembarski brojevi engleskih časopisa donose vesti o novim nevoljama u kojima se našao Sir Klajv Sinkler uoči velikog sajma časopisa PCW. Robert Maksvel je, naime, odustao od kupovine većine akcija Sinclair Research-a čim je saznao u kolikim se dugovima firma, zapravo, nalazi. Ovaj potez je mogao da označi trenutno bankrotstvo Sinclair Researcha da se nisu pojavili stari kooperanti — Timex, Sity Bank i Emi — koji su odlučili da Sinclairu pruže zajedničku pomoć. Ta pomoć je, naravno, došla uz određene ustupke, od kojih je najznačajnije spuštanje cene računara QL na svega 200 funti (do sada 400), uz najavu daljeg pada cene.

Ovaj potez je odlična potvrda više puta izrečene tvrdnje da je QL promašena mašina koja sebi nije uspela da pronađe tržište. Sa cenom od samo 200 funti, QL, koji ipak ima dosta memorije i procesor iz čuvene Motoroline serije 68000, konkuriše modelima iz ekonomske klase koji bi, po hardveru koji je u njih ugrađen, trebalo da budu daleko slabiji od njega!

Sniženje cene, suprotno očekivanju Sinclairovih kreditora, nije u prvom trenutku umanjilo zalihe QL-a. Dalja sudbina tvorca najpopularnijih kućnih računara na svetu je isključivo u rukama banaka, koje, uz blagoslove engleske vlade, treba da mu odobre nove kredite. Obzirom na Sinklerova iskustva u sličnim situacijama (firma Sinclair Research se već više puta spasavala sa same ivice bankrotstva), budimo optimisti!

D. Ristanović



Novi Epson

Tek što smo, u prošlim „Računarima", prikazali štampače Kaga KP810 i Canon PW080A, rekavši da su superiorniji od skupljih Epsonovih modela, firma Epson je pripremila odgovor koji se zove LX 80.

LX80 košta 242 funte+VAT, dok za dodatak koji omogućava pisanje na običnom papiru treba doplatiti još 20 funti. Osim 'Near Letter Quality' opcije i definisanja znakova, LX80 omogućava i neke sasvim nove stvari, kao što je automatsko desno uravnavanje ivice pri štampanju teksta — izgleda da će budući tekst procesori moći sve više da se oslanjaju na inteligenciju štampača. Iako je LX80 sa svojih 100 karaktera u sekundi sporiji i od Kage (140 cps) i od Canona (160 cps), reputacije firme Epson mu jamči veliki uspeh na tržištu.

D. Ristanović

Amstrad u Nemačkoj

Ako je u početku izgledalo da je „amstrad" dampingovao cenu sa namerom da obezbedi sigurno mesto svom inače izvanrednom prvencu, sa njegovim naslednicima — CPC 664 i CPC 6128 — jasno je da su cenu i kvalitet „Amstradovih" računara rezultat izuzetne poslovne strategije. Prema cenama na nemačkom tržištu, „Amstradov" kasetaš (verzija Šnajder) CPC 464 košta 899 DM, sa zelenim monitorom, i 1398 DM sa monitorom u boji. Amstrad sa ugrađenim disk-drajvom, CPC 664, prodaje se za 1496 DM ili za 1998 DM sa monitorom u boji.

Zanimljivo je da je dvostruko „jači" (128 K) i „zgodniji" (manji, poboljšana tastatura) CPC 6128 i relativno jeftiniji od svog prethodnika: 1598 DM sa zelenim monitorom i 2098 DM sa monitorom u boji. Kupcima u Nemačkoj Šnajder nudi i štampač daleko bolji od Amstradove verzije — NLQ 401 sa priključnim kablom košta 798 DM. Na sve ove cene kupci za izvoz imaju popust od 14 odsto.

Hakeri nisu usamljenici

Ako posedujete bilo koji računar, bilo koji modem (300/300, 1200/75 boda), telefon, 20 funti i puno para za plaćanje telefonskih računa, možete da iskušate svoje sposobnosti u igri MUD (Multi User Dungeon).

MUD je višekorisnička tekstualna avantura kojom upravlja VAX 750 pod VMS-om svakoga dana od 6 uveče do 8 ujutru (znate, hakeri uvek vole da rade noću), a preko vikenda po ceo dan. Kako sve to izgleda? Uplatite 20 funti početne članarine i još po 2 funte za svaki planirani sat igre i dobijate korisničko ime, lozinku i nadimak. Bilo kada u toku radnog vremena zamka možete da pozovete potreban telefonski broj i predstavite se računaru. Naći ćete se na mestu na kome ste prethodni put završili igru i moći da nastavite kretanje po zamku. Pri tom ćete, osim na raznorazne pauke, žabe, trolove i vešte nailaziti i na druge korisnike koji u tom trenutku igraju MUD; možete se boriti sa njima ili stupati u saveze radi ostvarivanja određenih ciljeva. Vaš rejting u igri će polako napredovati i na kraju ćete eventualno postati Čarobnjak (Wizard); od tog momenta možete sami da menjate arhitekturu zamka, otvarate nove prolaze i zagorčavate život ostalim igračima. U igri je stalno i „Arch—Wizard" koji pazi da u tom zagorčavanju života ne preterate!

Ako, i pored potencijalno astronomskih telefonskih računa, želite da igrate MUD, obratite se na telefon 2944-01-608-1173. Neka vaš računar i modem budu spremni jer vam, kao što i pristoji, neće odgovoriti ljudski glas!

D. Ristanović

Računari za škole

Komitet za prosvetu, kulturu, tehniku i fizičku kulturu SR Hrvatske nedavno je usvojio preporuke koje se odnose na minimalne uslove koje moraju da zadovolje kompjuteri za primenu u školama. Evo tih ***minimalnih uslova:***

1. Računar mora da bude domaće proizvodnje.

2. Tastatura mora da odgovara YUS rasporedu sa odvojenim tasterima za Č, Ć, Ž, Š i Đ. JUS, za slučaj da to niste znali, propisuje QWERTZ ili QWERTY tastaturu sa RETURN na najnezgodnijem mogućem mestu (sasvim desno u drugom redu, baš ispod tastera RESet i HOME), sa tasterom ESCape tamo gde je obično TAB, sa BRK pored ESC (kako biste ih lakše pomešali), sa znacima '-' i '>' na jednoj dirci i, da bi stvar bila posebno lepa, sa strelicama za pokretanje kurzora levo i desno od razmaknice — tek da ne biste mogli da editujete program jednom rukom.

3. Potrebna je kompatibilnost sa bar jednim svetski poznatim operativnim sistemom. Sa ovom se preporukom možemo bez rezerve složiti. Nevolja je, međutim, u tome što su preporučeni operativni sistemi CP/M i Apple DOS 3.3. Što se CP/M-a tiče, sve je u redu. Ali, preporučiti operativni sistem jednog računara koji je dobrano zašao u godine i koji, premda se dobro drži, više ne „gura" ni njegov proizvođač koji se okrenuo „mekintošu", pomalo je smešno. Ukoliko planiramo da nam se računari u školama ne menjaju svakih par godina, bilo bi razumno preporučiti samo jedan operativni sistem: MS DOS i IBM PC kompatibilnost. Treba se opredeliti za firmu koja dobija!

4. Osim bejzika, zahtevaju se logo i paskal. Obzirom na prethodnu preporuku, ova ne bi trebalo da zadaje probleme, jer su za CP/M, Apple DOS, MS DOS i sve ione poznatije operativne sisteme davno napisani interpreteri i prevodioci ovih jezika.

5. Dodatni harverski zahtevi su grafika visoke rezolucije, mogućnost animacije u koloru, ugrađeni džojstik ili miš kao i rad na nekom savremenom jeziku za nastavu kao što je pilot. Ova poslednja stavka se obrazlaže time što ima nastavnika koji ne poznaju programiranje a koji bi želeli da koriste kompjuter u nastavi. Sve u svemu, nastavnik koji ne zna programiranje će, da bi predavao deci koja znaju bejzik, paskal i logo, upotrebiti kompjuter, pre čega će naučiti pilot! Pre bismo rekli da takav nastavnik neće prići ni na puškomet kompjuterskom kabinetu.

6. Računar mora biti osposobljen za jednostavno povezivanje sa periferijom i drugim računarima, tj. imati ugrađen paralelni i serijski port prema međunarodnim standardima (verovatno RS232 i Centronics).

7. Računar ne sme biti preskup.

Nije teško pronaći računare koji zadovoljavaju uslove 2—7, ali oni ne zadovoljavaju uslov 1. Teško je ali moguće pronaći računar koji manje-više zadovoljava uslove 1—6 (jedan od takvih je Ivasimov Ivel Ultra), ali on svakako ne zadovoljava uslov 7. Sve u svemu, izgleda da će, i pored obilja računara koji su na svetu konstruisani poslednjih godina, naše škole još popričekati da se pojavi model koji zadovoljava ***minimalne*** *uslove za primenu.*

Dejan Ristanović

Računari na otpadu

Engleska, Švedska i druge zemlje koje se hvale visokim standardom mogu da se postide pred nama! Jer, samo kod nas na otpadima mogu da se pronađu novi i praktično neotpakovani računari!

Kako to? Našim firmama (ne treba im to mnogo zamerati) ne pada na pamet da kupuju domaće računare, a deviza i uvoznih prava nemaju. Rešenje se, naravno, nađe: neki vešt momak poseti neku susednu zemlju i na samo sebi znan način donese novog „spektruma", „komodora" ili, zašto da ne, IBM-a, a onda iznenada primeti da mu je taj računar bio nepotreban, pa reši da ga odnese u komision, naznačavajući cenu znatno veću od one koju je platio. Nekim čudnim slučajem upravo u tom trenutku pored komisiona prođe predstavnik firme kojoj trebaju računari . . .

Lepo, zar ne? Samo, komision uzima 22% provizije od ukupne cene plus još 10% poreza, a otpadi ukupno samo 10%. I tako naš novopečeni vlasnik računara primeti da njegova skupo plaćena naprava ne vredi ništa, pa je neotpakovanu odnese na otpad . . .

Dejan Ristanović

Video digitajzer

Nemačka firma Print Tehnik odnedavno proizvodi i prodaje video digitajzer za „Komodor 64". Mala plastična kutija priključuje se u „komodor" na zadnjoj strani računara i u nju se uključuje video kamera. Ovaj uređaj sliku dobijenu iz video kamere digitalizuje i prebacuje je u ekran visoke rezolucije računara. Odatle se slika može štampati ili snimiti na disketu ili kasetu i posle iskoristiti kao bilo koja slika visoke rezolucije. Glavni program, koji se dobija na disketi uz uređaj, omogućava da se slika digitalizuje i posle toga snimi ili štampa. Program je predviđen da radi sa dve, četiri ili svih šesnaest boja, a najbolja osobina mu je što je predviđeno štampanje na sve vrste printera koji se mogu priključiti na „Komodor", pa čak i na 1526 kod kojeg postoje neki problemi oko štampanja slika visoke rezolucije.

Proces digitalizacije traje oko četiri sekunde, posle čega se slika dobija na ekranu. Pošto kamera zahvata veću površinu od ekrana, možete pomerati ekran u svim pravcima i tako dobiti najbolju sliku. Na disketi se, pored glavnog programa, nalazi i nekoliko digitalizovanih slika, zatim nekoliko demonstracionih programa, kao i programi za rad sa četiri ili šesnaest boja. Moguće je digitalizovati bilo šta što kamera „vidi" — sobu, prirodu, lice i neku sliku ili fotografiju. Kao primer digitalizovano je dvadeset slika iz časopisa „Playboy". Cena uređaja u Nemačkoj je oko 400 maraka i može se nabaviti direktno kod proizvođača.

Ista firma prodaje i sasvim nov proizvod Voice Master 2. U reklami za ovaj harderski-softverski dodatak stoji da je to uređaj koji može da prepoznaje reči, rečenice ili tonove i da ih vrlo verno reprodukuje. Ukratko, dodatak koji govori i prima govorne naredbe. Pri tom se programski može predvideti koje reči prepoznaje vaš uređaj i kako na njih da reaguje. Uređaj, takođe, može da digitalizuje vaš glas, odnosno neku reč ili rečenicu. To, praktično, znači da možete da udete u sobu i ranije spremljenom računaru kažete: „molim te katalog diska" i „komodor", će vam dati sadržaj vaše diskete! Ako je to programom predviđeno, možete da zatražite da računar učita neki program, a računar može posle par sekundi da vam odgovori: „gotovo" ili da vam strogim glasom kaže „program se ne nalazi na ovoj disketi". Uz uređaj se dobijaju mikrofon, slušalice i disketa sa demonstracionim i pomoćnim programima. Cena uređaja je nešto manja od 300 maraka, a za sve informacije javite se na Print Tehnik. Nikolaistr. 2, 8000 Muenchen 40, tel: 089/368197.

V. Krstonošić

Spasavanje Acorn-a — drugi put

Septembarski Acorn User piše o drugom činu rešavanja Acornovih problema. Olivetti je, očigledno, i dalje voljan da ulaže novac a upotrebio je i uticaj kod banaka da isposluje dogovor u tri tačke:

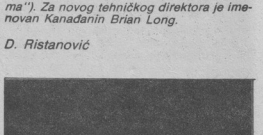
1. Glavni kreditori Acorna su se saglasili da otpišu polovinu dugova ove firme, s tim da im se neotpisani deo dugova isplati odmah.

2. Olivetti investira još 4 miliona funti u kupovinu akcija Acorna, čime poznata italijanska firma postaje vlasnik 80% akcija (ranije samo 49%). Osnivači acorna Hauser i Curry sada zajedno imaju jedva 15% akcija.

3. Acorn će, s obzirom da više praktično nije zadužen, dobiti od engleskih bankara dodatne kredite.

Acorn je, zahvaljujući tome, zaboravio na ideju o prodaji Acornsoft-a i o daljem otpuštanju radnika (sada kompanija ima 270 zaposlenih prema 480 u „zlatnim danima"). Za novog tehničkog direktora je imenovan Kanađanin Brian Long.

D. Ristanović



galaksija plus

Naš test

Mnogo novog hardvera . . .

Započinjući ovaj prikaz, teško je ne pri-setiti se dana od kojih nas dele dve godine: avgusta 1983. na našem stolu se našao jedan od prvih razvojnih primeraka „galaksije". Nikada nismo pokušavali da tvrdimo da je „galaksija" računar čije karakteristike izazivaju divljenje — to je, jednostavno, bio računar do koga možete lako da dođete i uz koji možete mnogo da naučite. Nadajući se uspehu akcije koju smo pripremali, mogli smo da pretpostavljamo da će za „galaksiju" biti napisano dosta programa i da će ona biti, u izvesnom smislu, i hardverski proširivana. Nismo, naravno, ni u snu mogli da pretpostavimo da će čitava stvar doći do tačke na kojoj se danas nalazi.

Prve verzije „galaksije" nisu imale čak ni port za proširenja! U toku meseci koji su sledili do definitivnog zaključivanja „Računara u vašoj kući", „galaksija" je pretrpela dve velike hardveske revizije i bezbroj izmena u softveru i tako ugledala svet u dobro poznatoj konfiguraciji: 6 K RAM-a, 8 K ROM-a (u prvobitno napisana 4 K je smešten osnovni operativni sistem i bejzik interpreter), softverski podržana grafika 64*48 i, naravno, port za ekspanziju. Preko 6000 naših čitalaca je, uz manje ili više problema, sagradilo ovakvu „galaksiju"; taj je odziv za sve nas koji smo sarađivali na projektu predstavljao veliku obavezu da radimo na daljim proširenjima našeg računara. I proširenja su došla: interfejs 1, memorijsko proširenje, EPROM programator, ROM 2, generator zvuka i, na kraju, fina grafika su poglavlja priče koja je krunisana projektom unapređenog modela nazvanog „galaksija plus".

Obzirom na značajno poboljšanje karakteristika, jasno je da novi model računara nije mogao da opstane sa samo 6 K RAM-a: „galaksija plus" ima 48 K dinamičkog RAM-a od kojih se, kao što vidimo iz memorijske mape, koristi 46. Dva kilobajta su, naime, morala da budu žrtvovana (ne može im se nikako pristupiti) da bi ih „poklopila" memorijski mapirana periferija: tastatura, kasetofon i hardver za prikazivanje slike.

Generator zvuka je dodatak za koji su se čitaoci „Galaksije" najviše zalagali u vreme kada je nah kompjuter još rastao. U tom trenutku se od njega moralo odustati, ali je „galaksija plus" bila prava prilika za ispravku: čip AY 3 8910 omogućava rad sa tri tonska kanala i dodatnim kanalom za generisanje belog šuma i dopunjen je malim zvučnikom koji se nalazi u kutiji računara. Generator zvuka je, na žalost, softverski praktično nepodržan, pa se muzika kontroliše jedino naredbom SOUND R.S. koja upi-

Lična karta

Mikroprocesor:	Z80A
Clock:	6144 KHz
ROM:	12 (14) K
RAM:	48 K (46 K pristupačno)
Tastatura:	59 tastera QWERTY sa slovima Č. Ć, Ž, Š
Ekran:	16 redova po 32 znaka
Editor:	Ekranski ili linijski
Grafika:	64*48 ili 256*208
Jezici:	BASIC interpretator, asembler
Aritmetika:	Pokretni zarez, 32 bita
Zvuk:	Tri kanala i beli šum
Kasetofon:	280 ili 1200 baruda
Disk:	U pripremi
Priključnici:	Televizor Kompozitni video Kasetofon
Proširenja:	Dva osmobitna porta (npr. štampač i opšti)
Proizvođači:	Zavod za udžbenik i nastavna sredstva, Obilićev venac 5/I Elektronika inženjering, Karađorđev trg 11, Zemun
Moguća cena:	140.000 dinara (?)

suje · broj S u registar generatora zvuka obeležen sa R. Sami ćete morati da pročitate tabele i zaključite koje brojeve treba staviti u koje registre da bi se čula željena melodija. Ukoliko se priučite ovakvom programiranju, moći ćete da kontrolišete i obojnice tona i tako proizvedete raznorazne zvučne efekte počev od zavijanja sirene pa do sinteze glasa. Obzirom da mikroprocesor, koristeći interapte, i dalje pomaže video stepenu pri generisanju slika, ne postoji mogućnost primanja signala za prekid od generatora tona. Ukoliko, dakle, želite da pišete igru praćenu muzikom, moraćete da produžite interapt rutinu Z80A i u tom produžetku podržavate generisanje zvuka (na taj način bi računar čak mogao da svira dok kucate neki program). One koji ne nameravaju da pišu komplikovane akcione igre će možda jedino nervirati činjenica da se pritiskom na BRK ton ne može prekinuti; na svu sreću, RESET je lek i za ovaj problem!

Moramo da kažemo da se rezolucija „galaksije" ne može povećavati nikakvim hardverskim ili softverskim dodacima koji ne bi promenili strukturu čitavog računara („Računari 1", strana 51) je rečenica koja opominje da treba biti obazriv u proračunavima barem kada su računari u pitanju. Jednostavnim zaobilaženjem generatora karaktera i pisanjem nove rutine koja opslužuje video „galaksija plus" je dobila grafiku visoke rezolucije 256*208 tačaka. Fina grafika je, kao što smo rekli, i dalje softverski podržana, što znači da Z80A troši 2/3 vremena na generisanje slike, a samo 1/3 na koristan rad; da je u „galaksiju plus" ugrađen neki video kontroler (npr. 6845 koji košta oko 6 funti, a koga koriste „amstrad", MSX i BBC), računar bi postao četiri puta brži, a procesor bi mogao da dobija interapte sa periferije i tako daleko racionalnije rasporedi svoje vreme! Razlog da konstruktori „galaksije plus" odustanu od video kontrolera verovatno leži u želji da se vlasnicima postojeće „galaksije" omogući jednostavno proširenje, kao i u potrebi da se očuva stoprocentna kompatibilnost (premda se zadovoljavajući stepen kompatibilnosti, verovatno, mogao postići i uz video kontrolor).

. . . i ne mnogo manje novog softvera

„Galaksija plus" ima 12 K ROM-a sa sistemskim softverom i opisima znakova: postojeći ROM-ovi 1 i 2 su sada ugrađeni u jedan čip 2764, dok su 2 K rutine koje opslužuju grafiku visoke rezolucije u EPROM-u 2716 (nazovimo ga ROM 3). „Galaksija plus", zbog kompatibilnosti sa osnovnim modelom, i dalje ima generator znakova upisan u drugi EPROM 2716 koji se nalazi u adresnoj mapi mikroprocesora. Prostor u ROM-u od sveta 12 K je, na žalost, morao da se troši na dvostruko ponavljanje istih stvari: u ROM-u 1 je interapt rutina koja crta sliku, a u ROM-u 3 druga a ipak slična rutina koja iscrtava sliku visoke rezolucije. U ROM-u 3 su morali da budu ponovljeni oblici svih karaktera jer mikroprocesor mora da ih ucrtava u bit mapu, a pri tom ne može da pristupa generatoru znakova. U RAM-u je, najzad, morao da bude rezervisan prostor od &2800—&29FF za sliku u niskoj rezoluciji i još 6.5 K za sliku visoke rezolucije, premda je jasno da se na ekranu u istom trenutku ne mogu nalaziti dve slike (treba, istini za volju, reći da se RAM između &2800 i &29FF koristi i u toku rada sa finom grafikom da se sadržaj ekrana ne bi izgubio po povratku u tekst mod).

Kako je moguće da su ROM 1 i ROM 2 ostali **apsolutno** neizmenjeni kada računar ima četiri nova tastera sa našim slovima?

Počinjući prikaz nekog novog računara, obično izlažemo kratku istoriju njegovog nastanka, govorimo o firmi koja ga je proizvela i o tržištu kome je namenjen. Prikaz „galaksije plus" ne možemo tako da započnemo: svi čitaoci našeg časopisa dobro znaju kako je „galaksija" nastala, kako je rasla i kakve je srećne trenutke i probleme donela onima koji su se odlučili za njenu samogradnju. „Galaksija plus" je, ako je posmatrate sa raznih strana, zaista sasvim nov računar, ali će ispitivački pogled u njoj otkriti dosta od naše (dve godine) stare „galaksije".

Kraljica na domaćem tržištu: „Galaksija" sa novom anatomijom (desno) i u novom ruhu (levo)

Zahvaljujući malom triku: tastatura je tako prevezana da kada pritisnete taster C softver u ROM-u 1 dobija informaciju da su pritisnuti tasteri C i SHIFT što on, jasno, registruje kao slovo C. Kada smo već počeli da odajemo male tajne „galaksije plus", reći ćemo da je u vezi sa oblicima naših slova bilo i softverskih mahinacija: u ROM 3 je, kao što smo videli, upisan generator znakova, pri čemu je, radi štednje, za svako slovo utrošeno po 7 bajtova. U taj prostor nisu mogle da budu upisane kvrge koje se nalaze iznad slova Č, Ć, Ž i Š pa je pribegnuto triku: pre nego što ucrta znak u bit mapu, računar proverava da li se možda radi o jednom od naša četiri slova; ukoliko na to dobije potvrdan odgovor, **najpre** iscrtava kvrgu u gornja dva reda tačaka; **a onda** ispisuje obično slovo C, Z ili S. Ukoliko, dakle, pokušate da predefinišete slovo Č u blanko, na ekranu će se pojaviti ' (apostrof)!

U hardveru „galaksije plus" se daju primetiti još neke vešto zamišljene sitnice koje su posledica odluke da se ne dira u ROM 1. Taster RESET, na primer, više ne povezuje NMI nožicu mikroprocesora sa masom već sa jednim od brojačkih kola u računaru. Time je postignuto da nemaskirani interapt bude onemogućen (da li to čujemo kako neko kaže da se nemaskirani interapt ne može onemogućiti? Ispravno je reći *ne može* **softverski** *onemogućiti*) u toku crtanja slike što znači da RESET više ne može da „zbuni" računar. Isti se efekat mogao postići da je instrukcija EI, koja se u ROM-u 1 nalazi na lokaciji &317 („farma"), pomerila malo dalje.

„Galaksija" i njen bejzik

Ostavljamo vam da sami prosudite da li tvrdnja „kompatibilnost je izgovor da se ništa ne radi" može da se primeni i na naše uslove. U svakom slučaju, odluka da ROM 1 i ROM 2 ne pretrpe apsolutno nikakve izmene će svakako obradovati svaku školu koja namerava da nabavi „galaksiju plus" kao jedini domaći računar za koji je napisan bar određeni broj programa: *apsolutno svaki* program pisan na bejziku ili mašinskom jeziku koji radi na „galaksiji" će bez ikakvih izmena raditi i na „galaksiji plus"; čak će i akcione igre biti jednako brze (spore?).

S druge strane, ovakva odluka može da razočara sve one koji su se klonili „galaksije" zbog „slabog bejzika koji je apsolutno nepogodan za primenu u školi". Neka autoru ovoga teksta bude dopušteno da, po prvi put, iskaže svoje mišljenje o ovome problemu: bejzik „galaksije" jeste i nestandardan i siromašan, ali to ne mora da znači da ga treba anatemisati kao kandidata za školsku primenu. Nije malo nas koji smo se sa programiranjem upoznali kupivši računar sa bednim bejzikom ili, čak, nekakav TI 58 koji se programira na simboličkom jeziku i ne poznaje ni jedan od elemenata savremenog strukturiranog programiranja i koji se danas sasvim fino snalazimo ne samo sa asemblerima i strukturiranim jezicima kao što su fortran 77, paskal i ada već i sa jezicima „novih generacija" kao što je prolog? Ako pronađete nekoga ko se sa programiranjem prvi put sreo radeći na paskalu, videćete da se takav sasvim lepo snalazi sa prologom, ali možda ima problema sa asemblerom. Ako, najzad, ikada budete pronašli jadnika koji je od malih nogu učio da programira na prologu, nemajte mnogo nade da ćete uspeti da ga naučite fortran, bejzik ili, ne daj Bože, asembler! Verujemo da će se svako ko je naučio da programira na siromašnom jeziku brzo (previše brzo!) privići na raskošan kompajler, ali će se onaj ko je navikao na moćan sistem naći u velikim problemima kada bude rešavao neki praktičan problem u skromnijim uslovima.

Bez obzira na ove filozofske rasprave, „galaksijin" bejzik ima nekoliko neprijatno slabih tačaka kao što su rad sa nizovima racionalnih brojeva i stringovima. Zbog toga se razmišlja o pisanju potpuno novog bejzik interpretera (procenilo se da postojeći nema smisla dalje krpiti) koji bi, zajedno sa operativnim sistemom, bio smešten u prvih 16 K adresnog prostora (promenom određenih džampera moguće je modifikovati memorijsku mapu „galaksije plus" tako da poprimi izgled prikazan na slici 2), dok bi ostatak memorije bio predviđen za RAM, uz mogućnost da posle dnjih 16 K zauzimaju pejdžovani ROM-ovi sa alternativnim jezicima. Nismo, međutim, sasvim sigurni da bi vredelo upustiti se u ovakav posao. Pošto je „galaksija" ostala jednako spora kao što je i bila, možda bi rad na razvoju novog bejzika bio uludo utrošen; šta vredi što možete da nazovete promenljivu BRZINA, ako će je kompjuter tražiti šest puta duže nego promenljivu B?

ROM 3 je održao visok standard koji su, u pogledu racionalnosti softvera, postavila prva dva ROM-a. U samo 2 K je smeštena nova rutina koja opslužuje video, proširenja bejzika koja, uz korišćenje izvanrednih algoritama, omogućavaju vrlo brzo crtanje linija, ekranski editor rađen po uzoru na „komodorov", te crteži svih slova koji sami zauzimaju prostor od &E552 do &E792

(&240 bajta). Onima koji ti znakovi nisu dovoljni ROM 3 omogućava slobodno definisanje karaktera, a onima koji se interesuju za animaciju mogućnost da za početak video memorije usvoji bilo koju lokaciju u RAM-u.

Pisanje i crtanje

„Galaksijinom" bejziku su, uz već pomenuto SOUND, dodate naredbe TEXT, GRAPH, PLOT, UNPLOT, DRAW i UNDROW. Po uključenju računara ekran biva obrisan (obzirom da RAM sada ima 46 kilobajta, pažljiv posmatrač će ovu operaciju primetiti), a onda se u njegovom vrhu pojavljuje poruka „*** GALAKSIJA PLUS ***" i dobro poznato READY. Umesto kurzora u obliku donje crtice (underscore), ispod poruke READY se pojavljuje pun karakter koji treperi dok je prompt u obliku znaka veće (>) nekuda išćezao (videćete i zašto). „Galaksija plus" se tako nalazi u tekst modu u kome je 99% kompatibilna sa starom „galaksijom" — razlika je jedino u tome što je umesto linijskog aktivan daleko moćniji ekranski editor. Da bi se uopšte radilo sa grafikom, treba izvršiti GRAPH a zatim se sa PLOT i DRAW crtaju tačke odnosno linije koje naredbe UNPLOT i UNDRAW brišu. Sve ove četiri naredbe imaju po dva argumenta koja predstavljaju koordinate neke tačke na ekranu. Kako linija može da bude određena samo jednom tačkom? Kao i kod mnogih drugih računara. DRAW 100,100 će spojiti *zadnju posećenu tačku* sa tačkom čije su koordinate (100, 100); ukoliko poželite da nacrtate liniju između tačaka (0.0) i (100, 100), izvršićete PLOT 0,0 (čime ste „posetili" tačku 0,0) a zatim DRAW 100, 100), izvršićete PLOT 0.0 (čime ste „posetili" tačku 0,0) a zatim DRAW 100, 100. Samo se po sebi razume da umesto DRAW 100, 100 možete da napišete DRAW I+J, 10*K ili nešto slično. Ukoliko ponovo poželite da radite sa starom, dobrom „galaksijom" ili da vidite razlike između starih i novih slova, izvršićete TEXT, a ukoliko želite da „ubijete" ekranski editor i tako postanete 100% kompatibilni sa osnovnom „galaksijom", izvršićete A=USR (&1000) a zatim, ako želite da budete dosledni, i mašinsko LD IY, &FD. Ponovno aktiviranje ROM-a 3 možete da postignete izvršivši A=USR(&E000).

Ne čini nam se, međutim, da ćete prečesto poželeti da isključite ekranski editor i odreknete se njegovih karakteristika: pritiskajući tastere sa strelicama pomerate kurzor (puni kvadrat) po ekranu i pozicionirate ga u okviru bilo koje linije. Svaki tekst koji dalje budete kucali će se prepisivati preko postojećeg sadržaja te linije koji će tako biti izgubljen. Kada završite kucanje linije pritiskate ENTER i naredba biva izvršena odnosno, ako ima linijski broj, unesena u program. Možete, naravno, da menjate i sadržaj ekrana ne pritiskajući ENTER čime nećete izazvati nikakve promene u programu; docnije možete da vratite kurzor u neku editovanu liniju i, pritiskom na ENTER, učinite da promene koje ste izvršili postanu trajne.

Mogućnost da se menjaju delovi programskih linija nije, naravno, dovoljna zakomforan rad: treba imati mogućnost za brisanje nekih slova i umetanje drugih. Za brisanje se brine taster DEL koji briše slovo ispod kurzora. Destruktivni backspace (taster koji bi pomerio kurzor za jedno mesto u levo *a onda* obrisao slovo ispod njega) ne postoji, ali vam on neće mnogo ni nedostajati: uvek možete da pritisnete levu strelicu i da otkucate ispravno slovo koje će biti uneto preko pogrešnog.

Za ubacivanje teksta je zadužena kombinacija tastera SHIFT i 0. Pritisak na njih pomera deo linije iza kurzora za jedno mesto udesno, dok se na mestu kurzora ukazuje slobodan znak. Pritiskom na REPT možete, naravno, da ponavljate ovu operaciju i tako u postojeću naredbu ubacite citavu reč. Nevolja, međutim, nastaje kada se linija koju editujete produži toliko da prede u sledeći programski red: „galaksija plus" će tada spojiti tekući red sa sledećim. Ukoliko ste to i želeli, odlično. Ukoliko niste, moraćete da pozicionirate kurzor na kraj smislenog dela naredbe, pritisnete DEL i tada REPT — sve dok „rep" ne bude obrisan. Verujemo da je ovaj svojevrsni bag ostavljen namerno, jer u ROM-u od 2 K posle svih sažimanja jednostavno nije moglo da se nađe prostor za dodatna testiranja. Nedostajao je i prostor za realizaciju „insert" moda koji autoru ovoga teksta užasno nedostaje; ekranski editor radi u takozvanom „overtype" modu što, kao što smo rekli, znači da se tekst koji kucate prepisuje preko postojećeg teksta. Verujemo da je u većini slučajeva daleko lakše raditi sa insert modom u kome se tekst ubacuje između kurzora i prvog sledećeg slova, što eliminiše potrebu za prečestim pritiskanjem tastera SHIFT i 0. Za komforan rad prilično nedostaje i opcija koju nudi većina drugih editora: pozicioniranje na kraj odnosno na početak linije pritiskom na SHIFT i desnu odnosno levu strelicu.

Ipak za školе

Nedostatak prostora je izazvao još jedan bag koji se javlja u toku izvršavanja programa: ukoliko napišete PRINT „UNESI DAN":: INPUT X$ a zatim otkucate SREDA, promenljiva X$ će dobiti vrednost „UNESI DAN?SREDA ". Bag nam otkriva ponešto o principu rada editora: negde u memoriji (zapravo iznad RAMTOP-a) je tabela u koju se upisana dužina svake linije na ekranu. Kada pritisnete ENTER, računar u bafer smešta tekst *od početka* tekuće linije pa do tačke koja je određena njenom dužinom. Prompt (upitnik odnosno, pri unošenju linija, znak (>) je, videli smo, morao da se izgubi kako se ne bi mešao sa vrednostima promenljivih.

Kada smo već kod bagova, primetili smo još po neki: ako u GRAPH modu popunite ekran linijama, od kojih se neke nalaze i u poslednjim redovima, svi budući skrolovi će ponavljati sadržaj poslednje linije teksta koja će se, osim toga, prepisivati u pretposlednju tako da zaostale crtice nikako (osim pritiskajući SHIFT i DEL ili izvršavajući HOME) nećete moći da prebrišete. Autoru ovoga teksta se dogodilo da se linija koju je editovao i koja je nosila broj 80 nađe na kraju programa, iza linije broj 750. Linija 80 je morala da bude obrisana mučnom primenom instrukcije BYTE jer je bejzik interpretator nije mogao pronaći! Nadamo se da će ovi bagovi koji nisu naročito ozbiljni ali koji mogu da budu neprijatni biti otklonjeni u trţišnim verzijama „galaksije plus".

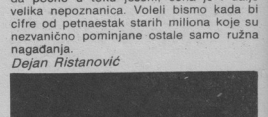
„Galaksiji plus" koju smo testirali bi mogla da se uputi još jedna ozbiljna zamerka: brzina komunikacije sa kasetofonom je premala za RAM od 46 K! Na svu sreću, praktično je razvijen ROM 4 u koji su upisane rutine za ubrzano snimanje i učitavanje *imenovanih* fajlova brzinom od 1200 boda. Preostali prostor u ROM-u 4 zauzimaju rutine za crtanje krugova i kosih elipsi i FILL procedura za popunjavanje zatvorenih površina.

Obzirom da postoji realna mogućnost da za „galaksiju plus" vrlo skoro bude razvijen disk interfejs baziran na čipu 1771, četiri kilobajta adresnog prostora između &000 i &FFF su predviđena za ROM 5 u koji treba da bude upisan DOS, operativni sistem za rad sa diskovima. U DOS bi trebale da budu upisane rutine za formatiranje i verifikaciju disketa, snimanje, brisanje i premenovanje fajlova i rad sa relativnim datotekama. Dalja softverska ekspanzija u vidu programskih jezika tipa torta ili, ako je za školske primene potreban, paskala je zasnovana na ugrađivanju alternativnih pejdžovanih ROM-ova.

Sve u svemu, „galaksija plus" je profesionalno dizajniran računar čiji su konstruktori, Nenad Đunjić i Milan Tadić, dosledno i marljivo sproveli koncepciju koju su sebi zacrtali. Po karakteristikama posmatrano, „galaksija" je i bez sumnje, najbolji domaći računar u svojoj i nekoliko viših klasa. Klasična u hardverskom dizajnu, „galaksija plus" je opremljena čudesno racionalno napisanim softverom, što joj je, paradoksalno, i najveća mana: da je sav rad utrošen na sažimanje ROM-ova 1, 2 i 3 u prostor od 10 K usmeren na proširivanje mogućnosti sistemskog softvera bez mnogo gledanja na prostor, računar ne bi bio ništa skuplji, a u našim bi mu uslovima bilo teško naći ravnog (posle bitke su, naravno, svi generali pametni: ko je tre dve godine mogao da predvidi da će EPROM 27128 (16 K) danas koštati samo 15 DM?). Kako stvari stoje, „galaksija plus" je dobra ali i nedovoljno uravnotežena mašina — veoma jaka po nekim karakteristikama i neprilično slaba po drugima. Kao takva, potpuno je neinteresantna za pojedince koji žele da se igraju i da koriste komercijalni softver i kojima je uvoz stranih kompjutera relativno otvoren. Za programere koji žele da se bave razvojem softvera, „galaksija plus" bi mogla da se pokaže kao privlačan izbor obzirom na veoma otvoren i dobro dokumentovan operativni sistem, veliku memoriju, dobar asembler i eventualni disk. Interesantna je i mogućnost razvoja edukativnih programa za samu „galaksiju plus", jer se ovakav softver može prilično povoljno prodati Zavodu za udžbenike i nastavna sredstva. Glavni prodor „galaksije plus" može da se očekuje u školama koje teško mogu da pronađu bolji računar sa više našim prilikama prilagođenog softvera i literature. Verujemo da je dalja sudbina „galaksije plus" neposredno diktirana odlukama prosvetnih vlasti.

Pokušali smo da ovaj prikaz dopunimo parametrima koje mnogi sa pravom smatraju najbitnijima: cenom računara i rokovima isporuke. To nam, na žalost, nije uspelo — iako bi proizvodnja nove „galaksije" trebala da počne u toku jeseni, cena je i dalje velika nepoznanica. Voleli bismo kada bi cifre od petnaestak starih miliona koje su nezvanično pominjane ostale samo ružna nagađanja.

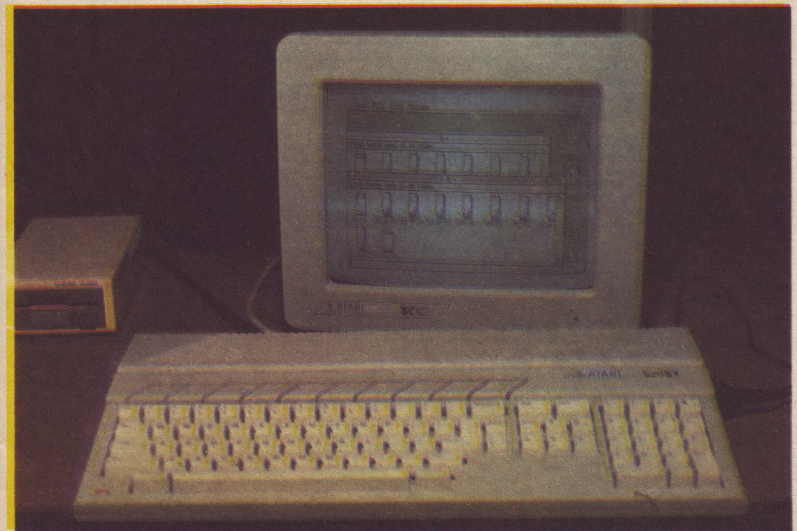
Dejan Ristanović



atari 520 st

Računari u izlogu

Ako želite da saznate koji je računar trenutno najbolji, raspitajte se kod osnovaca i srednjoškolaca koji još sanjaju o svom prvom kompjuteru! Oni će vas uveriti da sa izborom i kupovinom ne otežu zato što nemaju para nego zato što očekuju modernog princa na belom konju — računar sa šezdesetčetvorobitnim procesorom, najmanje šesnaest megabajta memorije i dva vinčester diska koji, u kompletnoj opremi, košta 99.99 dolara. Među takvim ljubiteljima računara „spektrum", „komodor" „amstrad" i svi kompjuteri koji koriste osmobitne mikroprocesore (posebno 6502) predstavljaju prezrene veličine. QL je „još jedna Sinklerova glupost", IBM PC je predmet nagonske mržnje (ako neko shvati zašto neka nam javi), a „prava stvar" je „atari 520 ST". Za ovaj izbor ima mnogo razloga: videćemo da je 520 ST računar izvanrednih karakteristika. Nevolja je jedino što ga na tržištu još nema, što znači da javnost zna samo za one njegove osobine koje proizvođači smatraju povoljnima!



Snimio: Zoran Tatar

Hardveristi bi rekli da je 520 ST (ili, popularno, „džekintoš") računar sa malo čipova: osim mikroprocesora, RAM-ova i ROM-ova koji su deo svakog kompjutera, 520 ST sadrži svega nekoliko standardnih kola i ULA čipova od kojih je najbitniji takozvani GLUE čija se uloga naslućuje iz samog imena (glue=lepak). Dizajniranje ULA čipova je, ako se očekuju velike serije, izvanredan način da se cena računara zadrži u prihvatljivim okvirima i da se dovoljno prostora u kućištu predvidi za buduća proširenja. No, podimo redom.

Više za korišćenje nego za programiranje: Atari 520 ST

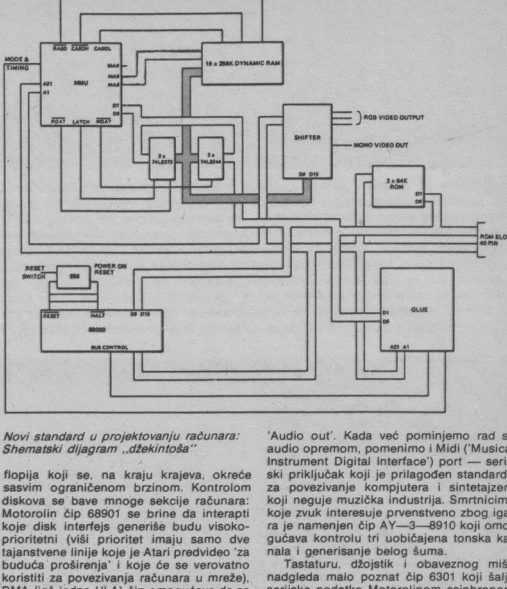
Ono 'ST' u nazivu kompjutera potiče od **Sixteen/ Thirtytwo** što znači da je srce džekintoša" Motorolin mikroprocesor 68000 (ovde nema Sinklerove lagarije koja se zove 68008) čija je arhitektura tridesetdvobitna, a komunikacija sa spoljnim svetom šesnaestobitna. Atarijevi inženjeri su odlučili za klok od 8 MHz, što možda i ne izgleda previše. Pokazuje se, međutim, da bi 520 ST mogao da bude pravi šampion u brzini jer je pozajmio jednu karakteristiku većih kompjuterskih sistema — modul za upravljanje memorijom ili skraćeno MMU (Memory Menagemant Unit).

Čudo zvano MMU

Potrebu za uređajem koji će upravljati memorijom su verovatno zapazili svi koji su pokušali da shvate kako radi 'spektrum' i slični računari. Mikroprocesor, sa jedne strane, mora da pristupa memoriji da bi obavljao svoj redovan posao dok video stepen podeset puta u sekundi crta sliku ispisujući na ekran sadržaj dela RAM-a. Stoga se pojavljuje konflikt: u nekim će trenucima i mikroprocesor i video stepen želeti da pristupaju raznim segmentima RAM-a što je, obzirom da računar ima samo jednu magistralu za podatke, potpuno nemoguće. Konstruktori „spectruma" i QL-a su mikroprocesoru zabranili da pristupa memoriji dok se slika crta, što značajno umanjuje brzinu rada računara. Konstruktori „džekintoša" su se dosetili nečem boljem: u toku write ciklusa podaci iz procesora se baferuju kroz dva čipa 74LS244, dok se u toku read ciklusa podaci lečuju kroz dva 74LS73, tako da ih procesor pokupi kada mu budu potrebni; data bus je tako slobodan za rad video kontrolera.

MMU se, jasno, koristi i za mnoge druge lepe stvari, kao što su proširenja RAM-a i ROM-a. Svi znaju da „atari 520 ST" u startu ima 512 K RAM-a koji je grupisan u dve kolone od po osam 256 kilobitnih čipova uklopljenih u memorijsku mapu tako da počinju od adrese 0. MMU ima deset multipleksiranih adresnih linija koje će omogućiti da se, umetanjem jednomegabajtnih čipova, RAM proširi do vrtoglavih 2 Mb. ROM, sa druge strane, zauzima vrh memorijske mape, završavajući se na &FEFFF (poslednja 64 K su rezervisana za memorijski mapirane ulazno-izlazne uređaje; ako vas upotreba ova 64 Kb posebno zanima, pročitajte „Veliku video predstavu" iz „Računara 6"). I pored ogromnog slobodnog adresnog prostora od 16 Mb, ROM-ovi su pejdžovani tako da zauzimaju 64 K adresa; MMU upravlja izborom ROM-a koji je u nekom trenutku potreban preko pet specijalnih linija koje poseduje već pominjani glue čip. U osnovnom modelu računara koristi se samo nulta linija koja selektuje interni šesnaestokilobajtni ROM sa osnovnim operativnim sistemom; ukoliko se GEM (o njemu docnije) bude našao u ROM-u, selektovaće ga linije 1 i 2. Linije 3 i 4 su odvedene do spoljnog podnožja za kartridže koji treba da omoguće povezivanje ROM čipova sa specijalnim programima (tekst procesori, razni programski jezici i kompajleri i slični) koji bi u svakom trenutku bili pristupačni korisniku.

Takvi ROM čipovi nisu, istini za volju, naročito potrebni: MMU se brine za neobično brzu komunikaciju sa periferijom, u našem slučaju disk jedinicama. 520 ST je opremljen 3·5" disk jedinicom na koju može da upiše standardnih 312 K podataka (uz dvostruku gustinu upisa (double density) ovaj bi se broj mogao gotovo udvostručiti ali izgleda da će „džekintoš" podržavati samo standardnu gustinu zbog kompatibilnosti sa „mekintoševim" formatom), a najavljen je i hard disk nepoznate cene. Što se samog računara tiče, MMU obezbeđuje prenos podataka brzinom od jednog megabajta u sekundi, ali će ta brzina biti višestruko umanjena karakteristikama samog



Novi standard u projektovanju računara: Shematski dijagram „džekintoša"

flopija koji se, na kraju krajeva, okreće sasvim ograničenom brzinom. Kontrolom diskova se bave mnoge sekcije računara: Motorolin čip 68901 se brine da interapti koje disk interfejs generiše budu visokoprioritetni (viši prioritet imaju samo dve tajanstvene linije koje je Atari predvideo 'za buduća proširenja' i koje će se verovatno koristiti za povezivanja računara u mreže), DMA (još jedna ULA) čip omogućava da se podaci sa diska prenose direktno u memoriju bez posredstva centralnog mikroprocesora, a čip YM 2149, zajedno sa 1772, selektuje površinu diska kojoj se pristupa, kao i potreban broj trake odnosno sektora.

YM 2149, osim podrške disk interfejsu, formira dva osmobitna paralelna porta i tri analogna izlaza. Jedan od portova predstavlja standardni Centronics interfejs za štampač, a drugi kontroliše RS232C interfejs. Tri analogna izlaza su povezana zajedno i odvedena do priključka koji će obradovati vlasnike pojačala i sličnih uređaja — 'Audio out'. Kada već pominjemo rad sa audio opremom, pomenimo i Midi ('Musical Instrument Digital Interface') port — serijski priključak koji je prilagođen standardu za povezivanje kompjutera i sintetajzera koji neguje muzička industrija. Smrtnicima koje zvuk interesuje prvenstveno zbog igara je namenjen čip AY—3—8910 koji omogućava kontrolu tri uobičajena tonska kanala i generisanje belog šuma.

Tastaturu, džojstik i obaveznog miša nadgleda malo poznat čip 6301 koji šalje serijske podatke Motorolinom asinhronom adapteru 6850 (dobro ga poznaju vlasnici BBC-ja i „electrona" obzirom da je odgovoran za razne zaštite softvera) koji ih pretvara u paralelne i opšti sa centralnim procesorom preko nisko prioritetnih prekida. Tasteri su interno raspoređeni u matricu 8*16, što znači da bi moglo da ih bude 128. Na tastaturi je, međutim, samo (!?) 95 dirki, dok su preostali povezani sa tasterima za paljbu na džojsticima i kontrolnim dirkama na mišu. Šematski prikaz hardverske organizacije „atarija 520 ST" je dat na slici 1.

„Atari" u radnjama

Nešto pre zaključivanja ovog broja „Računara", Byte, PCW i par drugih časopisa objavljuje da se „atari 520 ST" prodaje u Americi i u nekim specijalnim radnjama i sajmovima u Evropi. Kao što smo i pretpostavili, kompletan softver se prodaje na disketama, pa se uz kompjuter dobija TOS, GEM, BOS (programski sistem za poslovnu obradu podataka), GEM Write (editor teksta), GEM Paint (program za crtanje), bejzik i logo. Prvi utisci o tekst procesoru i bejziku nisu naročiti, dok je GEM Paint veoma hvaljen.

Cena „atarija 520 ST" sa jednom disk jedinicom i monitorom je u Engleskoj 760 funti (uračunavajući i VAT), a u SR Nemačkoj 2999 DM. Početkom septembra „atari 520 ST" je osvanuo ne samo u specijalizovanim radnjama za prodaju računara nego i u robnim kućama. Prijatna kupovina i — mnogo sreće na carini!

U svetu grafike

Video kontrolor omogućava rad u tri grafička moda: srednja rezolucija 320*200 tačaka, od kojih svaka može da bude obojena jednom od 16 boja iz palete od 512, visoka rezolucija 640*200 tačaka u četiri boje i veoma visoka rezolucija 640*400 u dve boje (na primer crno-bela slika). Obzirom da je 640*400/8/1024=31.25, od 512 K RAM-a treba odvojiti svega 32 K za video; za „džekintoša" niko ne može da kaže da korisniku ostavlja premalo slobodnog RAM-a! Da stvar bude posebno lepa, adresa početka video memorije može slobodno da se menja tako da, na primer, možemo da nacrtamo jednu sliku počevši od &10000, sledeću od &18000 a treću od &20000, a zatim, u samo jednom frejmu, zamenjujemo ove slike i tako stvaramo animaciju.

Hardverskom delu Atarijevog video interfejsa se, međutim, može uputiti i poneka zamerka. Pre svega, paleta je preko svake mere proširena (i na dobrom RGM monitoru teško biste primetili razlike između 512 nijansi), dok je izostao mod sa tridesetak boja koji bi vrlo dobro poslužio za razne profesionalne primene; računar čiji se RAM lako proširuje do dva megabajta ne bi upropastilo odvajanje 64 K za video. Osim toga, džekintoš se priključuje isključivo na monohromne ili RGB monitore što je za strance razumljivo: ako neko ima 700 funti za jedan profesionalan računar, još 100 za monitor ga neće upropastiti. Za nas je, međutim, situacija drućija: 100 funti ne predstavljaju baš malu paru, a monitor mnogo liči na televizor, što znači da je za carinike predmet velikog podozrenja . . .

Sa softverske strane, kontrolom videa se bavi GEM (Graphic Enviroment Manager) koji ujedno i komunicira sa korisnikom. Za sada je neizvesno da li će se GEM nalaziti u ROM-u ili će se učitavati sa diskete po svakom uključivanju računara: Atari je obećao da će softver upisati u EPROM(e), ali je i oprezno izjavio da će se on **možda** isporučivati na disku dok ne bude definitivno završen (znači li to da će se i „džekintoši" slati na doradu poput QL-a?). Pažljivi čitaoci ovoga teksta će primetiti da, s obzirom na brzinu „džekintošove" komunikacije sa periferijom, učitavanje GEM-a sa diska ne oduzima previše vremena, ali je ipak činjenica da je neprijatno tražiti sistemsku disketu po stolu i raznim kutijama kada god priđete računaru. Sa druge strane, na to traženje će vas verovatno naterati činjenica da se bejzik učitava sa diska, pa nećete mnogo razmišljati o tome gde je GEM upisan.

GEM nudi mnogo više od komandi MOVE, DRAW i CIRCLE. Ako ste na nekom sajmu videli Apple-ov štand, svakako znate za maksimu „ko ume da pritisne dugme taj ume i da radi sa „mekintošem" (odnosi li se to i na svakog petog nepismenog Jugoslovena?). Sličnu filozofiju je primenio i Atari: kada se računar uključi, operativni sistem iscrtava na ekranu razna lica koja se smeše i mršte u zavisnosti od toga da li ste ubacili dobar disk ili niste. Ovakav pristup će svakako privući mnoge Amerikance koji ne žele da se bave ničim osim svog posla, ali je za nas prilično odbojan: u Jugoslaviji računare i dalje kupuju uglavnom hakeri, a njih nervira kada ih kompjuter deset puta dnevno tretira kao budale. Vrlo je verovatno da će se upotreba svog budućeg softvera za 520 ST zasnivati na šetanju miša po stolu i izboru opcija iz mnogobrojnih menija. Korisno, ali ne i preterano uzbudljivo!

TOS ili CP/M

Osnovni operativni sistem „džekintoša" je TOS ili 'Tramiel Operating System'. Možda vam izgleda neukusno što je novi direktor Atarija nazvao operativni sistem po sebi, ali to samo znači da o njemu (i o sebi) ima vrlo dobro mišljenje. TOS, uzgred budi rečeno, ne predstavlja neki naročit novitet: radi se o popularnom CP/M koji je prilagođen Motoroli 68000. Ovaj izbor operativnog sistema je, po našem mišljenju, mudar akt očajanja. CP/M je, naime, dobar operativni sistem koji je primenjiv na računarima sa Z80 koji su opremljeni sa svega 64 K RAM-a. Motorola 68000 predstavlja, sa druge strane, izvanredno moćan mikroprocesor dopunjen ogromnim RAM-om; prisiliti ga da radi pod CP/M bi odgovaralo ugrađivanju peći na ugalj u Space Shuttle. Zbog čega je, onda, Tramijel pribegao ovakvom rešenju? I pored svojih optimističkih izjava, Tramijel po svoj prilici ne veruje da bi njegov računar mogao da dobije rat sa IBM-om koji je u svoj PC ugradio daleko slabiji Intelov mikroprocesor 8086 ali koji je opremljen nezamislivo velikim brojem programa. Naterati „džekintoša" da radi pod MS DOS-om bi značilo priznati poraz; naterati ga da radi pod CP/M-om označava jedino žrtvovanje karakteristika. Tramijel je, naravno, mogao da se opredeli i za izgradnju novog operativnog sistema koga bi sa nešto više prava mogao da nazove po sebi, ali za nešto ovakvo nije bio dovoljno hrabar (ili dovoljno lud).

Bilo kako bilo, da li CP/M treba da obraduje sve buduće vlasnike „džekintoša" u Jugoslaviji? Na to pitanje nije lako odgovoriti potvrdno: ako mislite da ćete moći da posetite prijatelje koji u nekom preduzeću rade na Iskrinom Partneru i da iskopirate (=piratujete) tekst procesor, bazu podataka i drugi „ozbiljan" softver, ljuto se varate. Program pisan za Z80 **sigurno** neće raditi na 68000 bez obzira na sličan operativni sistem. Da stvar bude još lepša, čak i razni CP/M računari sa istim mikroprocesorom najčešće nisu međusobno kompatibilni, ponajviše zbog formata disketa. Autori nekog CP/M programa će svoje remek delo bez previše problema prilagoditi bilo kojoj drugoj CP/M mašini (i tako duplirati prihode) ali će korisnicima teško uspeti da prepolove svoje rashode! Ako se, dakle, odlučite za 520 ST, pripremite dolare za kupovinu svog softvera koji vam je potreban.

Do sada smo u prikazima računara posvećivali posebnu pažnju interpretatoru za bejzik što kod džekintoša ne možemo da uradimo — bejzik, jednostavno, ne postoji u momentu kada pišemo ovaj tekst! Nepostojanje bejzik interpretera je jedan od dokaza da nije lako prilagoditi programe iz CP/M biblioteke koja je prebogata raznim verzijama bejzika novom TOS-u. Na beogradskoj demonstraciji u toku majskog Sajma tehnike „atari 520 ST" je, doduše, bio opremljen bejzik interpreterom, ali su taj program posetioci nazvali **dalekobilo bejzik**: poticao je iz „kamenog doba" razvoja ovog jezika i bio neopisivo spor, skoro sporiji od „spektruma". Niko, naravno, ne može da tvrdi da će se ovakav interpretator pojaviti i u konačnim verzijama, ali to nije sasvim nemoguće: treba da razumemo da, i pored svih izjava, Atari pretežno cilja na američko tržište na kome se računar ne meri kvalitetom bejzika i drugih programskih jezika (što se većine kupaca tiče, bejzik ne bi morao ni da postoji) već brojem i kvalitetom raznih aplikacionih programa koji mogu da se nabave. Sasvim je, dakle, moguće da će Atari sa džekintošem davati i neku disketu sa **dalekobilo bejzikom** dok će se, u bližoj budućnosti, na tržištu pojaviti nove verzije ovog jezika koje će kvalitetom (ali ne zasigurno i cenom) zadovoljiti programere.

Što je babi milo . . .

Zajedno sa bejzikom, uz „atari 520 ST" će se isporučivati i LOGO interpretator. Domaći hakeri, naravno, preziru LOGO: za šta će, zaboga, nekome naredbe koje šetaju puža po ekranu i crtaju njegov trag? Ako kažemo da logo podstiče lepo i strukturirano programiranje, ovaj će jezik biti još više omrznut. Sve u svemu, od jezika koje dobijete uz „džekintoša" se nećete previše odvajati, ali to nije ništa prema onome što tek treba da pročitate: uz računar **nećete** dobiti ni asembler, ni disasembler, ni debager ni bilo šta što je Pravom Programeru u životu potrebno (znate li ko je Pravi Programer? Čitajte u „Hakerskom manifestu" koga ćemo svakako objaviti u „Računarima 100"). Imati Motorolu 68000 u kući a nemati asembler? Pa, valjda će uskoro moći da se kupi!

Pripremajući ovaj prikaz potrudili smo se da konsultujemo što više stranih časopisa i primetili da u vezi sa „džekintošom" i dalje ima mnogo upitnika; u spekulacijama o nekom kompjuteru koga još nema na tržištu uvek smo skloni da idealizujemo njegove osobine, jer svi iz dna duše želimo kompjuter idealnih karakteristika. Takve spekulacije prave neviđenu reklamu modelu koji treba da predstavlja tehnološki skok na tržištu, Quantum Leap, što bi rekao Sinkler. Tako reklamirana mašina se, na žalost, jednom pojavi na tržištu i onda nastupa neizbežno razočarenje: ma koliko je računar dobar, on nikada nije onakav kakav bismo želeli da bude. Moćna firma kao što je IBM dopušta sebi da ne pisne o novom računaru sve dok on ne bude razaslat svim dilerima širom Sjedinjenih Država; manje firme, sa druge strane, moraju da privlače akcionare najavljujući unapred ono što bi želele da ostvare. Ako vam se čini da neosnovano impliciramo da će Džekintoš ponoviti sudbinu QL-a, prisetite se one narodne: „koga ujede zmija . . ."

Nezavisno od ovih strahovanja, pokušaćemo da sagledamo „atari 520 ST" u svetlu do sada poznatih karakteristika. Dizajn hardvera je izvanredan, do sada neviđen na jednom malom računaru. Što je najvažnije, hardverske karakteristike računara se daju poboljšavati do neslućenih razmera. Operativni sistem je, bez obzira na činjenicu da koristi samo deo mogućnosti procesora na koji je instaliran, sasvim solidan. Uz računar će se dobijati bejzik nepoznatog kvaliteta i ne naročito potrebni logo. Cena od oko 1000 dolara (2.999 DM sa crno-belim monitorom je niska kada se uzme u obzir ono što se za nju dobija. Ukoliko se, međutim, odlučite da kupite „atari 520 ST", očekujte dodatne izdatke od najmanje 300 dolara za kupovinu dobrog bejzik interpretera (ili fortran kompajlera), asemblera i disasemblera, tekst procesora i programa za rad sa bazama podataka. Očekujte, osim toga, i da će vas Atari snabdeti izvanrednim igrama po vrlo umerenim cenama. Pa, prijatna zabava!

Dejan Ristanović

ima li programera u avionu?

Računari u razgovoru

• Profesore, posle dosta dugog ćutanja profesionalnih programera i prepuštanja širenja računarske kulture zanesenjacima i trgovcima, pomalo smo iznenađeni da se i univerzitetski profesori sa priznatim naučnim radovima javljaju za reč na skupovima gde uglavnom govore oni za koje je i bejzik strani jezik. Stoga je naše prvo pitanje možda i neuobičajeno. Zašto ste prihvatili poziv da za čitaoce našeg lista govorite o računarima?

— Poziv sam prihvatio zato što mislim da se stručnjacima kod nas retko pruža prilika da govore o svom radu. Sredstva informisanja su puna prepričavanja tuđeg rada, obično netačnih prevoda tuđih članaka . . . Kako se numeričkom analizom bavim dve decenije, želim da zaštitim sebe i druge od neumesnih paušalnih optužbi da su stručnjaci „u ćeki". Nisam „u ćeki" i imam šta da kažem, mada se bojim da se to neće dopasti mnogim vlasnicima računara.

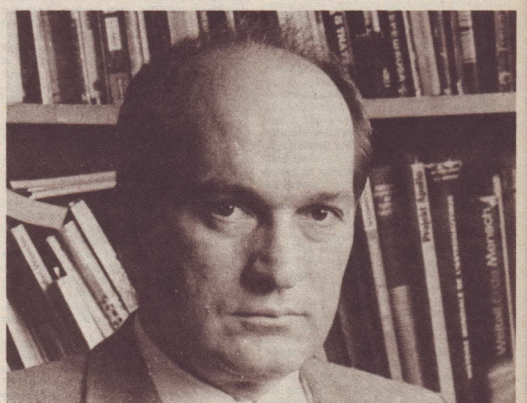
• Verovatno pri tom mislite na svoju tvrdnju da većina programera koji objavljuju programe u kompjuterskim časopisima i nisu programeri. To je prilično teška kvalifikacija. Možete li da je argumentujete?

— Zamislite program koji za većinu argumenata daje pogrešan rezultat. Možemo li pisca tog programa smatrati programerom? To je kao kad bi neki pilot većinu letova završavao havarijom. Neko napiše program za deljenje kompleksnih brojeva ili za izračunavanje modula i argumenta kompleksnih brojeva, pa za većinu argumenata taj program ne radi — šta na to reći? Ako taj i takav program nađe časopis za računare čak dva puta objavi — šta na to reći? A ako ni posle pola godine niko na to ne reaguje — šta na to reći?

• Da naši čitaoci ne bi bili u dilemi, recimo da se radi o programu objavljenom u časopisu „Moj mikro" i to prvo u slovenačkom izdanju, a zatim u izboru tekstova za prvi broj na srpskohrvatskom jeziku. Međutim, na vas primedbe nisu imuni ni neki programi objavljeni u našem listu, a da i ne pričamo o ostalim kompjuterskim časopisima koji tek fraze svoj stil. Ipak, amateram se redakcije za ovo snose samo deo krivice, jer mogu da objave jedino programe koje dobiju, a profesionalcima je lepod časti (ili je možda u pitanju novac) da ustupe svoje priloge. Da li biste, recimo vi, kada biamo vas zamolili da objavite neke od svojih programa kojima se popravlja aritmetika naših kućnih ljubimaca, poput ne malog broja vaših kolega rekli kako vam je žao jer nemate vremena za tako nešto, ili biste, na iznenađenje redakcije, pristali?

Zašto nel Imam puno profesionalnih obaveza, ali rado bih za čitaoce vašeg lista pripremio

• Zamislite program koji za većinu argumenata daje pogrešan rezultat. Možemo li pisca tog programa smatrati programerom? To je kao kad bi neki pilot većinu letova završavao havarijom.



Protiv pilota bez znanja:
Dr Dušan Slavić

programe iz oblasti kojom se bavim. Do sada to nisam učinio iz prostog razloga: to niko od mene nije ni tražio. Mislite li da je trebalo da tekst ponudim redakciji vašeg časopisa? Moram da naglasim da se u računarstvu uopšte ne radi o sukobu profesionalizma i amaterizma. Sredstva informisanja su prepuna priloga profesionalnih amatera, malo je amaterskih profesionalaca. Pojedinci su stručni za sve: od astrologije i biologije, do džunologije i šumarstva.

• Smatračemo, dakle, da možemo očekivati vaše priloge za „Računare". No, vratimo se mladim korisnicima računara. Da li svi oni uopšte treba da poznaju programiranje?

— Uporedio bih računarstvo sa vazduhoplovstvom. U tom poređenju: računaru odgovara avion, programeru pilot, a korisnicima računara — putnici aviona. Vaše pitanje analogno je pitanju da li svi putnici aviona treba da budu piloti, pa se nameće određen odgovor. Drugo je pitanje da li svi mladi treba da koriste računar. Odgovor je očigledno potvrdan. To su dva različita pitanja i ne treba ih zamenjivati. Od nekog korisnika računara, u zavisnosti od posla koji obavlja, može se očekivati da jednog dana prestane da samo poziva gotove programe. Obično se dogodi da poželi da i sam postane programer. To je ravno podvigu kao kada neko od putnika koji voli da leti avionom poželi da sedne za upravljač i proglasi sebe pilotom. Poslednje te hrabre odluke videće se posle prizemljenja aviona, odnosno u računatima koje će dati računar. Od pilota se očekuje bezbedan let, a od programera bezbedan program. On mora umeti da izbegava numeričke poteškoće i da, uz nedovoljno tačno predstavljanje i uzani opseg brojeva, dobija valjan rezultat. Valjan rezultat je brzo dobijen rezultat onoliko tačan koliko to dozvoljava tačnost računara.

• Ko je u stvari za vas pravi programer?

— Korisnik računara mora da poznaje svoj računar, a programer mora biti svestan svih manjkavosti osnovnog softvera i prevazići te slabosti. Test za prelazak korisnika računara u programera vrlo je jednostavan. Ako korisnik ume da načini algoritam i program za izračunavanje elementarnih funkcija i ako upoređivanje tih programa sa firminim daje dobre rezultate za sve kritične vrednosti argumenata — onda je smanjen broj korisnika računara i povećan broj programera.

Programer mora biti u stanju da rešava raznovrsne numeričke probleme. Od inverzije matrica, rešavanja nelienarnih jednačina do višestrukih integrala i rešavanja parcijalnih diferencijalnih jednačina. Da li dečaka koji o svemu ovome nema pojma treba smatrati programerom? Ako osnovac sekcira žabu — da li treba da sebe smatra hirurgom? Da li drugi treba da ga smatraju hirurgom zato što ima skalpel u ruci? Ja, naravno, nemam ništa protiv da taj dečak zaista postane hirurg, ali napominjem da školovanje za hirurga traje godinama — posle obaveznog školovanja.

• Znači, da bi neko bio pravi programer mora da doktorira matematiku?

— Ne radi se o tome da li neko ima diplomu pilota već da li ume bezbedno da leti i sleti. Programer mora da ume da načini programe za elementarne operacije i elementarne funkcije — i to potpuno pouzdane programe, tačne do poslednjeg bita. To je moguće. To mora umeti da uradi. Ako to ume niko ga neće pitati za

Na jednom skorašnjem sastanku posvećenom računarstvu vanredni profesor beogradskog Elektrotehničkog fakulteta dr Dušan Slavić osvetlio je problem korišćenja računara iz jednog novog ugla. Osim što predaje Programiranje i Numeričku analizu na fakultetu i vodi više kurseva na postdiplomskim studijama, dr Slavić je autor oko sedamdeset objavljenih naučnih radova i programer koji je prepravio i poboljšao sistemski softver mnogih velikih i malih računara i tako omogućio da se i oni koriste za precizna računanja. Njegova videnja, iznesena na dvadesetogodišnjem Iskustvu, imaju otuda veliku težinu. Zamolili smo da za „Računare" iznese svoja shvatanja o programiranju i računarima i pojavama koje ih prate.

• Imao sam prilike da vidim programe koji baš nikome neće trebati ili će značiti grdnu dangubu.

diplomu, već će mu poveriti komplikovanije poslove.

Ljubitelj šaha zna kada je mat. Ako ljubitelj šaha ne zna da dâ mat — nećete ga smatrati šahistom. Za pravila šaha dovoljno je nekoliko minuta, ali ako neko zna pravila to još ne znači da je postao šahista. Pravila programskog jezika mogu se naučiti vrlo brzo, ali znati programski jezik ne znači znati programirati.

• Najveća nevolja u radu mladih na računaru je što oni zapravo ne znaju šta da načine od svog poznavanja programskog jezika. Šta biste preporučili onima koji žele da postanu profesionalni programeri?

— Imao sam prilike da vidim programe koji baš nikome neće trebati ili će značiti grdnu dangubu. Tužno je što je u pisanje tih programa uložen ogroman trud. Učenici bi trebalo da već iz školskog gradiva crpe ideje za svoje programe. Ogromno je polje njihovog rada. Transformacija matematičkih teorema za potrebe računara je upravo počela. Ništa nisu propustili, ovo je tek početak.

Budući da želim da se broj programera radikalno poveća moram da istaknem: rešavanje numeričkih problema na računaru je neuporedivo zanimljivije od bilo koje video-igre. Rezultat rada programera dodaje se rezultatima drugih programera. Veliki projekti se razlažu na gotove programe i samo se dopisuje deo koji nedostaje. Sam rad na računaru je izuzetno zanimljiv već i zato što treba mašini izdati naredbe koje ona nikad neće razumeti, ali će izvršiti upravo ono što se od nje traži. Za mlade je to korisna situacija. Za svaki nesporazum sa računarom, ako su dobri osnovni programi, kriv je korisnik računara. Time se razvija samokritičnost. Već zbog toga je neprihvatljiva zabrana uvoza računara ili dozvola uvoza samo računara-igrački.

• Zakijučili smo već da će samo mali broj korisnika računara postati programeri. Svi, međutim, vole da ga koriste za igranje. Kako gledate na video-igre?

— U našoj analogiji upotrebi računara isključivo za igre odgovara avion parkiran pored piste u kome je unutrašnjost preuređena u luna-park. Za takve avione nisu potrebni piloti, piloti se bave letenjem. Postoje jedino nakazni svemirci ili nešto drugo, u zavisnosti od veštine dizajnera,

• Mi, razume se, zahvaljujući trgovcima kojima je omogućeno da imaju monopol na jugoslovenskom tržištu, kupujemo to što nam nude i u situaciji smo da ako sami ne napišemo bolji softver koristimo ovaj š-te kategorije.

13/ima li programera u avionu

što treba tamaniti. Nikada nisam imao vremena da igram neku video-igru. Već poodavno nisam dete, a još nisam podetinjio. Moj sedmogodišnji sin se igra na računaru, kada je računar slobodan. On uticaju video-igara na nežnu dečju psihu ne pitajte mene — nisam za to stručan.

• U svojoj praksi koristili ste više velikih računarskih sistema projektovanih za naučnotehničku primenu, za koje je logično da imaju bolju tačnost od kućnih računara koji koštaju desetak puta manje od jednog njihovog terminala. Kako stoji stvar sa tačnošću tih velikih i skupih računara?

— U svetu poznata firma u svom priručniku za naučno orijentisan računar IBM 1130 piše da njen proizvod za aritmetičarenja za veliku vrednosti modula argumenta daje pogrešne rezultate (veće od PI/2 ?!). Zar nije bilo lakše i časnije korigovati program? Programe za sve elementarne funkcije ovog računara morali smo da izmenimo. Bilo je moguće učiniti sistemske programe i tačnijim i bržim i kraćim.

Pojedine firme su svesne teškoća pravljenja pouzdanih osnovnih programa. Na primer, firma OLIVETTI za računar M20 ima četiri osnovne računske radnje u dvostruko] tačnosti, ali ne i elementarne funkcije, a potrebne su komercijalne opservatorije načinio sam pouzdane i tačne programe elementarnih funkcija. Neki računi naprosto zahtevaju veću tačnost. Tako je za opisivanje kretanja Meseca, na primer, potrebno 16 značajnih decimalnih cifara.

Ne manje poznata firma DIGITAL svoj računar VAX-11/751, nedavno instaliran na Elektrotehničkom fakultetu u Beogradu, snabdela je brojnim programima za elemntarne funkcije realnog i kompleksnog argumenta bez ikakve korektno. To je savremen računar od koga se s pravom očekivalo da ima pouzdane programe. Međutim, pokazalo se da su programi za elementarne funkcije ispod svake kritike. Na primer, ako je argument korektan, računar javlja „nekorektan argument" i prekida dalji korisnički program. Ako argument nije korektan — računar ne javlja grešku i daje pogrešan rezultat. Gde je moguća velika tačnost računar ne daje ni jednu tačnu cifru rezultata itd. Na simpozijumu koji organizuje svake godine DECUS (Društvo korisnika DIGITAL-ovih računara) prošle godine sam dokazao da su pogrešni svi programi za elementarne funkcije. Iako simpozijum većim delom finansira firma DIGITAL, rad mi je objavljen.

Zamislite muke korisnika takvog računara koji se uzda u firmu i program i dobija pogrešne rezultate. Korisnik greške traži u svom programu i ne sluti da su osnovni programi koje poziva pogrešni. Na žalost, mali broj korisnika računara je svestan mogućnosti računara i ograničenosti osnovnih programa.

• Koliko je meni poznato, Amerikanci su otišli najdalje u osvajanju mogućnosti brzog i tačnog računanja, a iz vaših reči se može zaključiti da oni nisu sposobni da pristojno reše čak ni računanje elementarnih funkcija. Da li sam je u velikoj zabludi ili u vašim rečima postoje i neki skriveni parametri?

— Onima kojima je poznato u kakvom su položaju tehnološke kolonije — a ne budimo u zabludi, mi smo danas, kada se radi o računarima, upravo to — ti parametri i nisu skriveni. Amerikanci su još kako sposobni da naprave dobru aritmetiku za svoje računare, ali nama, kao i istočnim zemljama prodaju za skupe pare sistemski softver koji rade njihovi pripravnici. Mi, razume se, zahvaljujući trgovcima kojima je omogućeno da imaju monopol na jugoslovenskom tržištu, kupujemo to što nam nude i u situaciji smo da ako sami ne napišemo bolji softver, koristimo ovaj š-te kategorije.

• Isključivo zalaganje za samo jedan programski jezik neodoljivo me podseća na reklamu: „Udišite samo čist ozon — nikako običan kiseonik".

• Posle ovog odgovora ne mogu a da vas ne upitam još nešto. Ne tako davno imali smo prilika da na televiziji gledamo svečano otvaranje novog Računarskog centra na Elektrotehničkom fakultetu i tada je rečeno da se radi o modernom sistemu ISKRA-DELTA, a vi pričate o nekakvom „vaseu". Mislimo li na isti sistem?

— Da, mislimo na isti, ali posle je ISKRA-DELTA na ovaj sistem stavila jedino nalepnice ne vidim da treba da ga smatramo njenim računarom i on će za mene i one koji ga koriste uvek biti VAX 11/751.

• Ako ste ovako počastili VAX 11/751, sa umetničkim imenom ISKRA-DELTA, šte ii cete tek reći o aritmetici ličnih računara?

— Kada se govori o tačnosti i brzini programa obično se setim računara „spektrum". Elementarne funkcije tog računara računaju se na skrata izuzetno načlo. Tako rade samo programeri ser Klajv Sinklera. Većina elementarnih funkcija izračunava se ekonomizacijom potencijalnih razvoja. Prvi deo ekonomizacije je transformacija potencijalnog razvoja. u Čebiševljev razvoj i odbacivanje suvišnih članova tog razvoja, u skladu sa tačnošću računara. Drugi deo ekonomizacije je transformacija skraćenog Čebiševljevog razvoja u ekonomizirani potencijalni razvoj. Za ovaj drugi deo ekonomizacije Sinklerovi programeri očigledno nisu čuli. Njihovi programi sadrže izračunavanje vrednosti Čebiševljevog razvoja, a ne vrednosti ekonomiziranih potencijalnih razvoja. Zato programi „spektruma" traju dva do tri puta duže nego što je to potrebno. Zamislite avion koji od Beograda do Dubrovnika leti nekom čudnom stazom Beograd—Zagreb—Skoplje—Dubrovnik. Uveravam vas da za to nema nikakvih stvarnih razloga. Nedoučenost programera je jedini razlog — programi mogu biti i tačniji i brži i kraći. Kvadratni koren na ovom računaru ne izračunava se po algoritmu starom preko dve hiljade godina koji koriste drugi računari. „Spektrumov" program poziva funkcije LOG i EXP. Ta ekstravagancija košta dosta vremena. Pored toga, za male vrednosti i za velike vrednosti argumenata relativna greška

• Koliko poznajem softver naših prvenaca — on je ili preuzet sa hardverom, dakle obično loš, ili je domaći, dakle još gori.

rezultata je preko sto puta veća nego što je dopušteno. Dozvoljeno je jedino da poslednji bit mantise bude nesiguran.

• U ime „galaksijinih" konstruktora i onih koji su je izabrali za školski računar zahvaljujem vam što ste pričali o „spektrumu". Ali, šalu na stranu, pošto je za numeričku analizu vrlo važna tačnost, vi biste prilikom izbora računara prevashodno poklonili pažnju ovoj karakteristi-

ol. Koliko se o njoj može saznati iz prikaza novih tipova računara u našim kompjuterskim časopisima?

— Prikazivači računara obično zaboravе, ili misle da to nije važno, da navedu dva osnovna podatka o računaru: tačnost predstavljanja realnih brojeva i opseg brojeva. Ta dva podatka svaki numeričar mora imati u vidu kada piše program bilo u svojstvu korisnika računara ili programera. Dodajmo ovim kriterijumima i treći: brzinu. Nikako nije svejedno koliko će dugo da rade osnovni programi. Neki vrlo hvaljeni numerički algoritmi zahtevaju veoma veliki broj interakcija. Zamislite avion koji načini na hiljade krugova nad aerodromom pred sletanje. Dobro je da se rezultat dobije za života programera.

• **Niste baš nežni prema projektantima računara. Postoji li uopšte kućni računar koji biste mogli da koristite za svoj naučni rad?**

— Za razradu numeričkih algoritama programeru je potreban računar sa bar devet značajnih cifara. Moji dosadašnji računari imali su 10, 12, 13 i 17 značajnih cifara. Razume se, nije svima neophodna tolika tačnost, ali ja poredim međusobno algoritme i ova mi je osobina veoma značajna. Meni najviše odgovara računar na kome mogu, ne smanjujući raspoloživu memoriju, zameniti firmine programe svojim, a za zorno prikazivanje greške potreban mi je ploter. Za sada nemam disk jedinicu, već učitavam programsku podršku sa trake, ali to ne usporava bitno moj rad, jer radim neprekidno više časova.

• **Da li možda koristite računar SHARP MZ 731?**

— Pogodili ste. Nisam imao nameru da ga direktno pomenem — imam jake razloge za to. Taj računar je osobit po tome što ima samo 4K ROM-a i 64K RAM-a. Njegov S-BASIC ne koristim, jer ima samo 9,5 značajnih cifara i nije naročito razvijen. Umesto njega koristim Hu-BASIC, koji ima 17 značajnih cifara. Za sva numerička istraživanja to je prava stvar.

• **A mogu li se na sličan način modifikovati i bejzici na drugim kućnim računarima?**

— Mogu, ali uz znatno više truda.

• **Sudeći po tome što koristite Hu-BASIC, vi se implicitno zalažete za bejzik. Recite nam, molim vas, eksplicitno, kakvo je vaše mišljenje o bejziku?**

— Uz sve poštovanje prema Fortranu IV, na kome sam programirao vrlo dugo, kao i prema Fortranu 77, koji koristim za veće projekte, najviše deo vremena ipak radim na Hu-BASICU, jer ga imam na kućnom računaru. Za moj rad su sasvim dovoljni tipovi podataka koje on nudi, a strukturirano programiranje o kome se kod nas toliko priča u poslednje vreme, mada odavno nije u prvoj mladosti, za mene predstavlja luksuz, jer znači strašno rasipanje memorijskog prostora. Uostalom, novi bejzici imaju u odnosu na prve verzije ovog jezika znatno bolje mogućnosti i mogu se sasvim udobno koristiti. Isključivo zalaganje za samo jeden programski jezik neodoljivo me podseća na reklamu: „Udišite samo čist ozon — nikako običan kiseonik".

• **Da li Jugoslavija može i treba da proizvodi sopstvene računare?**

— Pitanje je analogno sa: da li Jugoslavija treba da proizvodi avione. Odgovor je jasan. Ako se radi o vojnim avionima ne samo da treba nego i može, pa i proizvodi sa uspehom; ali ako je reč o putničkim avionima tipa DC-10, odgovor je: ne. U računarstvu je situacija slična. Ako zahvaljujući stranoj tehnologiji i sopstvenoj pameti možemo sebi i svetu da donudimo nešto jeftinije i bolje u nekoj klasi računara — onda ne samo da treba nego bismo to i morali da činimo. Ali uopšte nisam siguran da su dosadašnji pokušaji uspešni. Koliko poznajem softver naših prvenaca — on je ili preuzet sa harderom, dakle obično loš, ili je domaći, dakle još gori. Za harderer ne pitajte mene, ja sam samo diplomirao na Elektrotehničkom fakultetu i harderom se bavim samo koliko moram.

nastavak na str 64

Kako postati kompjuterski hakersko gluvarenje

Kako to stvarno izgleda biti kompjuterski ekspert? Evo ovako:

— Kad god negde zakasnite, možete da se izvadite pričom kako ste rešavali neki važan softverski problem (vidi Rečnik). Ljudi će vam uvek verovati jer su to za njih, kao i za vas uostalom, uglavnom „španska sela". To će vam, između ostalog, dati i određeni „imidž".
Vaš šef na poslu nikad pojma nema šta vi, zapravo, radite. Kad god pokušate da mu to objasnite, on će imati važan sastanak na drugom mestu.
— Niko nikad nema pojma gde ste. „Verovatno je u Sali" (s velikim S — odnosi se na prostoriju u kojoj su razbacane metalne sandučine koje nazivate „moj Sistem" ili „moja Mašina", s velikim S i velikim M, naravno) ili „Sad je tu nosio neki listing". Za to vreme, vi, naravno, rešavate ukrštene reči ili se bavite drugim adekvatnim intelektualnim poslom.
— Od vas se neće tražiti da bilo šta zaista proizvedete; nikad.

I, najbolje od svega, računari će biti široko isključeni iz vašeg života. Ovo je naročito važno u današnjem svetu, u kome se od svakog, od čistačice do profesora filozofije, očekuje da koristi, ili razmišlja o korišćenju računara u svom poslu.

Ovaj priručnik vam obezbeđuje osnovni vodič u planiranju kompjuterske karijere. Uputstva su jednostavna i korisna jer se baziraju na prostom, više puta u praksi proverenom principu: količina vašeg uspeha kao kompjuterskog eksperta obrnuto je proporcionalna vašem interesu za računare i kompetentnosti i znanju u njihovom korišćenju.

Gde tražiti posao

Osnovno pravilo u traženju posla je da izbegavate preduzeća koja zaista mnogo koriste računare u svom poslu. Verujte mi na reč, problemi koje ta preduzeća pokušavaju da reše svojim računarima uglavnom su dosadniji i nezanimljiviji i od samih računara. U takvim preduzećima, čak postoji mogućnost, mada zanemarljivo mala, da će, nakon enormnih troškova koje su preko raznoraznih samoupravnih foruma odobrili za nabavku računarske opreme, neki ljudi početi da primećuju da se baš ništa ne događa, i početi da pitaju zašto.

Ako već morate da radite za nekog korisnika računara, evo nekoliko uputstava kako da izaberete pogodno preduzeće:

— Ono bi trebalo da ima najzastereiju računarsku opremu u zemlji (to je, doduše, teško jer je u tome kod nas konkurencija vrlo velika). Ovo pokazuje stav rukovodećih i samoupravnih struktura u preduzeću prema računarskoj tehnologiji.
— Knjigovodstvo i računovodstvo, u tom preduzeću, trebalo bi da vode ljudi koji su jedom nogom u penziji. Oni se neće mnogo uzbuđivati zašto ne predlažete kompjuterska rešenja za oiakšavanje njihovog posla.
— Računarska aktivnost bi trebala da bude utopljena u tokove birokratije i administriranja sa reputacijom da nikada pravovremeno ne odgovara na zahteve stvarno produktivnih delova preduzeća.

Preduzeće koje odgovara ovim uslovima nije teško naći u našoj poslovnoj svakodnevici.

Nikada ne prihvatajte posao u sektoru ili službi koja ima sopstveni računar sa kojim želi da reši sopstvene probleme. Poželjan naziv sektora ili službe u kojoj bi trebalo da želite da radite treba da glasi SAOP (sektor ili služba automatske obrade podataka) ili ERC (elektronski računski centar) ili RIS (razvoj informacionih sistema) ili tako nekako. Njegove osnovne karakteristike su da on koristi računare za svoju sopstvenu zabavu i nastoji svim silama da obeshrabri sve druge da ih i oni koriste.

Najbolje zaposlenje za kompjuterskog eksperta je radno mesto kod nekog od domaćih proizvođača ili „proizvođača" računara. Ima nekoliko razloga zašto je tako. S jede strane, sami proizvođači računara u principu vrlo malo koriste računare u sopstvenom poslu. S druge strane, totalno neznanje u oblasti računara će vam jako pomoći da vrlo brzo postanete jedan od rukovodilaca u takvom preduzeću. I najzad, svi proizvođači računara imaju spiskove čudnovatih i ekscentričnih tipova koji obožavaju njihova metalna čudovišta i znaju apsolutno sve o računarima budući da sa njima provode sve raspoloživo vreme. Bez takvih ljudi bi, inače, proizvođači računara brzo mogli da zabrave bravu. Ako uspete da se ne nađete na takvoj listi u vašem preduzeću, niko neće ni obraćati pažnju na vas.

Radna okolina

Kompjuterski ekspert mora jako da obraća pažnju na svoju pojavu i utisak koji ostavlja na druge. Oko toga postoje neka fundamentalna pravila kojih se treba strogo pridržavati:

— Održavajte svoj sto konstantno zatrpanim raznim listinzima, šemama i literaturom. Ovo daje utisak da ste uvek u centru zbivanja, pretrpani idejama i problemima koje temeljito rešavate.
— Pišite podsetnike o svemu i posvuda pravite zabeleške. Upotrebljavajte kratke rečenice sa bombastičnim izrazima i ne brinite ako nemaju nikakvog smisla. Svi će verovati da su te zabeleške dubokoumne i da su pune skrivenih značenja.
— Uvek izražavajte zabrinutost i rezervisa-

ekspert

Računari nisu obavezno i bezuslovno dobra stvar. Iako postoji ogroman broj kompjuterskih igara koje vas mogu navesti da provodite dane i noći buljeći u ekran, u korišćenju računara ima relativno malo praktično mogućnosti, naravno, ako se izuzme kompjuterski kriminal.

Međutim, kompjuterski kriminal zahteva izvanredno poznavanje računara i njihove prateće opreme, stalno i veoma pažljivo praćenje brda stručne literature, veliku maštovitost i vrlo precizno planiranje, a to, zaista, ne može baš svako. Umesto toga, postoji jedan mnogo lakši način da dođete do novca baveći se računarima — da postanete kompjuterski ekspert. Ovo ne donosi toliko para koliko i kompjuterski kriminal, ali je zato mnogo manje rizično i zahteva neuporedivo manje rada.

Ilustracija: Miša Marković

nost. Nikad ne budite zadovoljni nijednom tehničkom odlukom koju je vaše preduzeće donelo.

— Obećavajte bilo šta. Rukovodeće garniture lako zaboravljaju. U svakom slučaju, za opravdanje uvek postoje „objektivne okolnosti".

Kako ne izabrati specijalnost

Svaki kompjuterski ekspert mora da izabere da li će biti opšti ekspert ili specijalista. U osnovi, ne možete početi karijeru kao opšti ekspert, jer za to treba dosta vremena, mada, srećom, vrlo malo napora. Da biste postali opšti ekspert, morate, prvo, puno vremena da provedete razmatrajući različite i raznovrsne projekte (to, naravno, ne znači da bilo šta stvarno morate da imate u vezi sa njima).

Dakle, postati opšti ekspert je dugoročan cilj. Kako izgleda opšti ekspert? Pa, kao prvo, on ne zna gotovo ništa o gotovo svakoj stvari u vezi sa računarima, ali zna da koristi stručni žargon u primerno korektnim rečenicama. Na primer: „Pre nekoliko godina, generisali smo makroe na sličan način, ali smo tada koristili ADSAC na MRC8600. Međutim, pronašli smo da je onda linkovanje rutina nepouzdano".

Iz prethodne izjave proističe još jedna važna činjenica: o čemu god da je reč, opšti ekspert se ponaša kao da je on to već odavno pronašao i uradio, pa mu je to sve sada i pomalo dosadno. Takođe, dobar opšti ekspert obično vrlo autoritativno tvrdi kako je sve ono što on ne zna potpuno nevredno da se zna, ili je, čak, i opasno. To čini opaskama, kao što je: „Naravno, vi to možete da pokušate da rešite, ali ne znam da bi od toga moglo da bude bilo kakve koristi". Isto tako, često se koristi i drugojačiji pristup: „Oni nemaju nikakvu teorijsku potkovanost u tim stvarima. Sve je to „ad hoc" i može samo loše da završi".

Biti opšti ekspert je dobar način da se izbegne etiketa o tehnološkoj zaostalosti, kojom kompjuterski eksperti stalno nastoje da diskreditiju jedan drugog. Oni se, pri tom, služe ovlašnim primedbama, kao što su: „On misli da je to High-level programiranje, zato što su programi pisani na četrnaestom spratu" ili „Poslednja osoba koja je njegove ideje uzimala za ozbiljno bio je Miloš Obrenović".

Opšti eksperti su uvek u dobroj situaciji, jer primaju visoke lične dohotke i teraju tako do penzije. Vremenom, oni se učvršćuju u svom položaju, obeshrabrujući mlade stručnjake, ulivajući im osećanje da je sve vredno već odavno urađeno, ili da je suviše teško da se uradi.

Posle razgovora sa savetnikom za razvoj softvera u svom preduzeću, jedan moj kolega sa fakulteta rekao mi je, sa tragičnom dozom razočaranja, da su stvari „tako kompleksne, da mu se ponekad čini kako ništa nije vredno da se radi".

Ovaj moj kolega je u osnovi pogrešio: ništa JESTE vredno da se radi. Ta sposobnost da se mladi, nadobudni stručnjaci muvaju tamo-vamo i čini opšte eksperte dragocenim i nezamenljivim u preduzećima.

Kako izabrati specijalnost

S obzirom da kao što je već rečeno, status opšteg eksperta dolazi tek posle nekog vremena, novi kompjuterski ekspert mora da izabere svoju specijalnost. Nemojte se obeshrabriti stravičnim spiskom mogućih specijalnosti, budući da pravi specijalista o svojoj oblasti mora da zna samo toliko da na sastancima ume da sastavi nekoliko prostih potvrdnih rečenica. Pri tom je bitno obratiti pažnju na odnos subjekta i objekta, na primer:

pogrešno: Razvoj aplikacija olakšava otvorenost operativnog sistema.
tačno: Otvorenost operativnog sistema olakšava razvoj aplikacija.

Ima boljih i gorih specijalnosti. Ove druge uglavnom su vezane za hardver, elektronsku komponentu računarskog sistema, i to zato što su ti rezultati rada merljiva kategorija. Ova činjenica je dovela do takvog razvoja računarske tehnologije da značaj hardvera sve više opada, dok enormno raste značaj firmvera i softvera.

Dakle, većina dobrih specijalnosti nalazi se u softverskoj oblasti, a najbolje specijalnosti su u vezi sa softverskim proizvodima za koje niko stvarno ne veruje da su potrebni. Među najbolje specijalnosti spadaju:

— Dizajn operativnih sistema. Ne postoji nijedan poznati način da se uspe u ovoj oblasti, ali ne postoji ni jedan poznati način da se ne uspe u njoj. Ova oblast odnosi se na kreiranje programa koji čine računare lakim i efikasnim za korišćenje. Najbolje je kreirati glomazne operativne sisteme, jer takvi sistemi nikad nisu sasvim završeni, niko pojma nema šta zapravo rade, a i niko ih inače i ne voli.
— Projektovanje sistema. To znači predviđati kakve računare korisnici žele da imaju. Niko neće verovati vašim predviđanjima. Niko neće znati šta bi sa njima čak i ako im poveruje, jer još ne postoji proveren način za realizaciju zahteva korisnika u proizvodnji računara. Ovo je veoma dobra specijalnost pošto, u stvari, niko i ne želi računare za koje tvrdi da ih želi.

Sva istraživanja korisničkih zahteva pokazuju da ljudi žele računare koji su pouzdani, laki za instaliranje, laki za korišćenje i pogodni za proširivanje. To jednostavno nije istina. Radna mesta mnogih ljudi upravo se baziraju na okolnosti da su računari nepouzdani, teški za instaliranje, teški za

Mali rečnik kompjuterskih eksperata

Ada — e. nešto čemu jedino treba da znate **ime da biste bili kompjuterski ekspert.** Korisno u rečenicama kao: „IBM je taj problem vrlo lako rešio, ali on je imao adu".

Automatizacije obrade podataka — ž. uvođenje računara radi povećanja efikasnosti poslovanja uklanjanjem svih onih sa kojima biste voleli da proćaskate u toku radnog vremena.

Bag — e. (bug) majušna kreatura koja živi u kompjuterskom programu i čini da ovaj pogrešno radi. Aktivnost „ćišćenja programa od bagova" prestaje kad se ljudi umore od traženja, a ne kad se ovi stvarno i uklone.

Bejzik — e. (Basic) apsolutno veštačka tvorevina koju sa gnušanjem odbacuju svi oni koji programiraju na drugim programskim jezicima. Hvaljen i obožavan od onih koji na njemu zarađuju velike pare držeći raznorazne „male škole Bejzika".

Dokumentacija — ž. ono za šta uvek kasnije požalite što niste pripremili na vreme.

Hardver — e. (hardware) sve ono na računarskom sistemu što možete da šutnete nogom kad se isnervirate.

IBM — ž. ogromno mitsko čudovište kojim kompjuterski eksperti plaše miade stručnjake: „Ako ne budeš miran i dobar, doći će IBM da te prožderе".

Kompatibilan računar — ž. isti sa onim drugim, kao jaje sa jajetom. Dođuše, on je glup, a onaj drugi pametan; njemu od onog drugog ništa ne pristaje; on ne razume ništa što bi onaj drugi razumeo; ali, kad Vam kažem, on je potpuno kompatibilan.

Literatura — ž. uputstva prevedena sa japanskog na engleski za čoveka koji zna samo srpskohrvatski.

Nezavistan paket — e. (machine independant package) programski paket koji ne radi ni na jednom računaru.

On-lajn — e. (on-line) ideja da bi ljudsko biće uvek trebalo da bude na raspolaganju računaru.

Paskal — e. (Pascal) programski jezik koji je dobio ime po čoveku koji bi se prevrnuo u grobu da zna za to.

Performanse sistema — ž. odnosi se na brzinu kojom računarski sistem radi. Ili bi radio pod određenim okolnostima. Ili se govorka da je radio negde u Ljubljani pre oko tri meseca.

Prioritet — e. (priority) stvar od važnosti za korisnike velikih računarskih sistema. Često se izražava kao relativni prioritet pokazujući da korisniku nije važno koliko mu se program brzo odvija, samo ako se ne odvija sporije od programa nekog drugog korisnika.

Sistem-programer — ž. osoba apsolutno potrebna radnoj organizaciji, kad je reč o problemima koje treba da reši. Apsolutno nepotreban radnoj organizaciji, kad je reč o ličnom dohotku koji bi trebao za to da dobije.

Sistemska podrška — e. (system support) grupa kompjuterskih eksperata čiji se posao sastoji u tome da vam objasne zašto od računarskog sistema ne možete da dobijete podatke koji vam trebaju.

Softver — e. (software) delovi računarskog sistema koji se ne vide, pa ih ne možete rastaviti da bi videli šta zapravo rade, ali bez kojih računar ne bi mogao ništa da radi. Kao takav, po definiciji, produkt rada kompjuterskih eksperata.

Statistički modell — ž. skup matematičko-statističkih formula koje pokušavaju da objasne zašto su stvari sve gore i gore.

Strategija — ž. dugoročni plan radne organizacije, čiji neuspesi ne mogu biti razmatrani, sve dok njegov kreator ne napusti radnu organizaciju.

Legenda:
e. — engleski izraz
ž. — žargon kompjuterskih eksperata

korišćenje i gotovo nemogući za proširivanje. Zbog toga proizvođači računara prave nove računare kako se njima sviđa, a onda ih reklamiraju kako su prilagođeni potrebama korisnika.

Možda najbolja specijalnost za kompjuterskog eksperta je obrazovanje. To je oblast sa velikim brojem pod-specijalnosti:

— Obučavanje ljudi koji ne znaju ništa, ali bi želeli da nauče da koriste računare. Ovo izbegavajte.
— Obučavanje ljudi koji ne znaju ništa, ali neko drugi ih tera da nauče. Ovo nije tako loše. Predavaćete ljudima koji su vam slični.
— Obučavanje kompjuterskih eksperata koji žele da prate računarsku tehnologiju. Držite se ovoga. Tu ste među prijateljima.

Osnovne strategije u karijeri

Nalazimo se u muškom WC-u velikog računarskog centra. Direktor se okreće od pisoara i ugleda kompjuterskog eksperta. Ekspert se nalazi pred susednim pisoarom, okrenut leđima, i tako ostaje za sve vreme razgovora. Ovo se smatra vrhunskom umetnošću.

Direktor: „Pa, Sale, šta misliš o ovom našem novom projektu?"
Ekspert: „Ja sam pomalo zabrinut."
Direktor (Još se osmehuje): „O? Zabrinut? Zašto?"

16/kako postati kompjuterski ekspert

— Nemojmo im dozvoliti da shvate šta možemo činiti sa ovom spravom, jer će nas inače, naterati da radimo osam časova u kancelariji . . .

Ekspert: „Nisam siguran da su sigurnost i zaštita podataka rešeni kako treba. Znate, u prošlom broju časopisa Computer World, Brogler i Kevendiš su pisali o toj temi . . ."
Direktor (Osmeh nestaje): „Naravno, dizajn sam prepustio projektnom timu. Mislim, ja u njih imam puno poverenje . . ."
Ekspert: „Znate, čini mi se da postoje problemi u adresnoj tabeli. Izgleda da je moguće generisati adresu zaštićenog podatka. Kevendiš tvrdi da . . ."
Direktor „Ali projekat je kako treba, zar ne? Misliš li da je to veliki problem? Do đavola, dve milijarde, a prokleta stvar nije kako treba! Slušaj, ako to ne radi, možeš li da preuzmeš ispravku na sebe?"
Ekspert: „Čekajte, to u osnovi radi. Samo nije elegantno. Ono što Brogler i Kevendiš tvrde je da . . ."
Direktor: „Hajde onda, laku noć, Sale, i nemoj da ostaješ predugo."

Dakle, da li smo ovde imali scenu u kojoj je dobar kompjuterski ekspert, stvarno zainteresovan za kvalitet, razgovara sa poslovnim birokratom? Ne, ovde smo videli neobično elegantno korištenje prodavanja zjala.

Prodavanje zjala sugeriše da je kompjuterski ekspert vrlo važan stručnjak, kome je puno stalo do kvalitetnog softvera, čak i u vrlo privatnim momentima, ali je suviše nestabilan da bi se na njega oslonio u pogledu konkretnog posla.

Vrlo važan momenat u odnosima rukovodilac-ekspert je da se nikada ne bude pozitivan. Kompjuterski ekspert nikada ne sme da kaže ništa kao „Ovo izgleda dobro" ili „Čini mi se da je sasvim u redu". Rukovodeće strukture ne poštuju eksperte koji su prijatni i ljubazni. Tada ih smatraju običnim personalom, ili misle da ovi nešto kriju. Ipak, budite oprezni! Takođe je istina da rukovodeće garniture često ne trpe nesiganje. Prava mera bi bila diskretna, ali neefikasna opozicija. Talasajte, ali ne pravite neprilike!

Primetite da ekspert nije napravio nikakav komentar o samom projektu, već se zadržao na opskurnom detalju. To je dobar potez. Iskren i kritičan komentar bi pokazao da ekspert pojma nema čemu pomenuti projekat služi, kome uopšte treba i zašto mu treba.

Važno je, takođe, primetiti da pravi kompjuterski ekspert nikada ne bi odigrao opisanu scenu, kada Brogler i Kevendiš ne bi bili stvarni ljudi, koji su stvarno napisali članak o projektovanju informacionih sistema. Naravno, uopšte nije potrebno da:

1. Kompjuterski ekspert pročita taj članak, ili razume o čemu se tu, zapravo, radi;

2. Kompjuterski ekspert ima bilo kakvu predstavu o tome kakve veze ima rečeni članak sa novim projektom u njegovom predvečeru;

3. Kompjuterski ekspert zaista ima potrebu da toliko dugo stoji pred pisoarom.

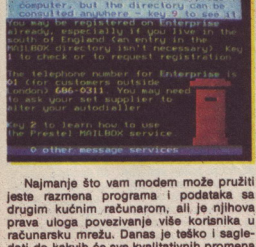
Ako se dobro izvodi, prodavanje zjala je jedan od najelegantnijih strategijskih poteza u stvaranju karijere kompjuterskog eksperta. Ipak, za njegovo korišćenje potrebna je velika veština, jer uvek postoji izvesna, mada najčešće zanemarljivo mala, doza rizika. Naime, vi tu, kobojagi, demonstrirate visoko stručno znanje pred ljudima koji bi, takođe, takvo znanje trebalo da poseduju. Međutim, nemojte se nipošto bojati. U najvećem broju slučajeva, ma kakvu glupost da izvalite, ako to učinite sa autoritetom, ljudi će se složiti sa vama, jer bi u protivnom došli u opasnost da se otkrije kako o toj materiji znaju čak i manje od vas.

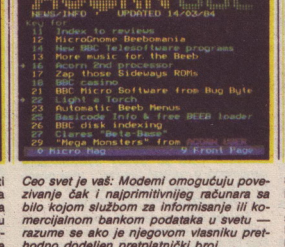
Cilj prodavanja zjala je da direktor izvuče određene zaključke o kompjuterskom ekspertu: da nije stabilan, da mu sudovi nisu izbalansirani, da, možda ima problema s bešikom. Njegovo znanje i stručnost savršeno ne dolaze u pitanje. To čini kompjuterskog eksperta izvanredno posebnom osobom — koja se nikad ozbiljno ne kritikuje, ali se ni ne smatra mogućim izvorom produktivnog rada. To je idealan status kome treba da težite dok ste u cvetu karijere.

Aleksandar Demel
Vrhunski kompjuterski ekspert

svet na dlanu

Obrada podataka koji se prikupljaju sa udaljenih perifernih jedinica, tzv. telecomputing, odavno se vrši na našim profesionalnim računarskim sistemima. Imamo, takođe, i bogato iskustvo u sprezanju više računara u mrežu, čime se omogućava izmena programa i podataka ne samo među računarima istog tipa nego i između onih različitih proizvođača i mogućnosti. Tako i korisnik jeftinog mikroračunara spregnutog u mrežu raspolaže programskom podrškom nekog velikog i moćnog računara kao i njegovom tehničkom opremom — može svoje informacije da šalje na njegovu disk jedinicu ili brzi paralelni štampač. Sve ove primamljive mogućnosti danas su na raspolaganju svakom vlasniku kućnog računara snabdevenog specijalnom perifernom jedinicom, MODEMOM, koja omogućava povezivanje njegovog računara preko telefonske mreže sa bilo kojim drugim računarom ili informacionom mrežom.





Najmanje što vam modem može pružiti jeste razmena programa i podataka sa drugim kućnim računarom, ali je njihova prava uloga povezivanje više korisnika u računarsku mrežu. Danas je teško i sagledati do kakvih će sve kvalitativnih promena u svakodnevnom životu dovesti u bliskoj budućnosti ova mogućnost. Osvrnućemo se, zato, samo na neke aspekte koji su već prisutni u SAD, Velikoj Britaniji i drugim zemljama koje su neki korak ispred nas u informatičkoj revoluciji.

Sve Informacije na dugme. Za većinu vlasnika modemi omogućavaju, ako ništa drugo, ono bar pristup MICRONET-u 800.

Ceo svet je vaš: Modemi omogućuju povezivanje čak i najprimitivnijeg računara sa bilo kojom službom za informisanje ili komercijalnom bankom podataka u svetu — razume se ako je njegovom vlasniku prethodno dodeljen pretpletnički broj

Micronet je sekcija British Telecom Prestel servisa, koja svojim pretplatnicima nudi neku vrstu kompjuterskog magazina putem telefona. Postoje hiljade stranica koje pokrivaju sve teme koje biste našli u štampanim časopisima. Sadrži stranice sa novostima, preglede, tehničke savete, softver (ponešto i besplatno), klupske biltene i reklamu.

Neke sekcije Prestela su, kao i sam Micronet, imovina zatvorenih grupa i stoga se naplaćuje. Ali ostala Prestelova mikroračunarska baza podataka, Viewfax 258, raspoloživa je svima sa dosta novosti, telešopinga i telesoftvera.

Osim Prestela, u Evropi, kao i SAD, sve se brže granaju mreže računarskih sistema koje pokreću privatni entuzijasti, pa se i kod nas, ukoliko nas ne iznenadi neka šira društvena akcija, može uskoro očekivati bar neka mreža spektrumovaca ili komodorovaca.

Savremeni šoping bez novca i korpe. Za obavljanje kupovine više se ne mora odlaziti u prodavnicu. Dovoljno je da svojim računarom uspostavimo vezu sa računarom trgovine. Naš posao svodi se na izbor artikala iz kataloga. Izbegavamo i odlazak u banku po novac — on se na osnovu porudžbine direktno uzima sa našeg tekućeg računa. Na sličan način možemo rezervisati avionsku kartu ili mesto u hotelu.

Elektronska pošta. Mejlboks (mail-

box) omogućava dopisivanje bez trošenja hartije, mastila i poštanskih maraka. Svoja pisma upućujemo kroz telefonsku mrežu u memoriju računara primaoca, a i sami s vremena na vreme „prelistamo" šta su nam prijatelji pisali.

To su neke od mogućnosti koje možemo koristiti već danas. Ali uskoro bi trebalo da imamo i pogodnosti poput sledećih:

Idealno obučavanje. Šta mislite o učenju isključivo kod kuće? Ne ustajući iz kreveta možete pratiti predavanje na ekranu svog televizora, ali ne kao uobičajen TV program, jer se i od vas sve vreme očekuju reakcije. Na ovaj način u domove hiljade učenika mogu se useliti predavanja podržana najsavremenijim nastavnim sredstvima, ni video disk za ovakve uslove neće biti suviše skup., a za nadarene će uvek po usvajanju propisane količine znanja biti na raspolaganju čitave biblioteke — videoteke koje danas nemaju ni najbogatiji univerziteti. Profesori će pak, imati mnogo bolji uvid u zalaganje i rezultate svojih učenika i moći će više pažnje da posveti najtalentovanijim.

Radno mesto — u kući. Konačno, ono što bi najviše moglo da oslobodi čoveka, je mogućnost da i svoje profesionalne obaveze obavlja iz kuće. Povezemo se izjutra sa računarom u svojoj radnoj organizaciji, obradimo poslove koji su pripseli i pošaljemo rezultate natrag u firmu. Možemo raditi kada nama odgovara, štedimo vreme, benzin i nerve, a učinak je bolji nego sadašnjom organizacijom posla.

Sve u svemu, budućnost nam se osmehuje ali . . . U čoveku leži neki vrag koji ga ne pušta tako lako da ide napred. Sadašnja papirnata administracija teško će se pomiriti sa ovim promenama koje se neminovno moraju desiti. Sem toga, izmislićemo sigurnо i rafiniranije vrste „računarskog" kriminala, strašićemo svoje prijatelje lažnim porukama, pokušavaćemo da krademo svoje i tuđe banke i trgovce, varaćemo svoje profesore elektronskim putem i, razume se, balkanski ćemo zabušavati na svojim radnim mestima i preko računara. I ako to budu sve „strahote" za koje će nam služiti kućni računari vezani preko modema u računarske mreže, naše perspektive uopšte nisu loše.

Uloga modema

Kada šaljete poslovno pismo, uobičajeno je da se počne nekom frazom (kao Poštovani druže Petroviću ili Dragi prijatelju itd.) i završi se na odgovarajući način (S drugarskim pozdravom, na primer). Za većinu zvaničnih pisama očekivaćete i neku vrstu odgovora, ako ste poslali ček za neku robu, očekivaćete obaveštenje o isporuci i upustvo za njenu upotrebu kada stigne. Računar to takođe očekuje i uloga modema je da osigura da će računar to i dobiti.

„Dragi Z80A . . ."

Kada je reč o vašem računaru, poruka se sastoji od bajtova podataka, koji za drugi računar predstavljaju poruke. Stoga se svaki bajt mora otvoriti, zatvoriti, a zahteva i neku vrstu odgovora. Otvaranje se vrši „startnim bitom", a zatvaranje „stop bitom". Naravno, kada je bajt završen, računar očekuje sledeći, ali će prethodno poslati odgovor slanjem njegove kopije računaru-predajniku da bi se proverila tačnost.

Uz malu pomoć satelita: Da bi se olakšao život na pustom ostrvu dovoljni su, očigledno, jedan računar, jedan modem i jedan telefon

Odgovor stiže na drugoj frekvenciji, tako da računar koji šalje mora čekati na odgovor, što se u žargonu zove „puni dupleks" (full duplex). Češće se koristi „polu dupleks" (half duplex) veza, što podrazumeva da se informacije prenose samo u jednom smeru, a zatim samo u drugom smeru, dok simpleksni prenos (isključivo u jednom smeru) nije pogodan za naše potrebe.

Ali to nije sve. Računar-prijemnik takođe treba da ima garanciju da je pristigla tačna informacija, jer postoji verovatnoća da su je „šumovi" na liniji pokvarili. Garanciju predstavlja dopunski bit umetnut ispred stop bita koji se bira tako da dekadna vrednost celokupnog koda uvek bude paran broj (ili neparan, ako sistem takav očekuje). Ovaj ekstra bit zove se „bit parnosti" (parity bit) i jedno je od sredstava kojima se povećava pouzdanost prenosa informacija. Vidimo da je za prenos jednog jedinog alfanumeričkog znaka za koji je u ASCII kodu dovoljno sedam bitova, pri asinhronom stanju neophodno čak deset bitova. Naravno, u jedinici vremena preko telefonske linije se može poslati samo jedan bit, pa se svaki bajt koji se u okviru samog računara prenosi paralelno, pre slaganja mora razbiti u bitove, odnosno mora se izvršiti transformacija paralelnog u serijski prenos. I ovaj posao moraju da obave modemi, obično uz pomoć standardnog RS 232 interfejsa.

Većina sistema prispeli osmi bit uopšte neće uzeti u razmatranje jer će ga tretirati kao bit parnosti, što znači da na ovaj način možete poslati samo kodove sa vrednošću do 127. Verovali ili ne, to ne pravi nikakve posebne poteškoće. Kodove iznad 127, spektrumove službene reči, na primer, interfejs „dekodira" u ASCII kodove. Tako, ako na interfejs dođe vrednost 245, što predstavlja kod za PRINT, umesto njega biće dalje poslati kodovi za slova P, R, I, N i T.

Kada interfejs „usortira" informaciju, ona se šalje u sam modem koji je oblikuje za slanje telefonskim linijama. Koliko brzo — zavisi od posla koji je modem trebao da uradi i standard za koji je projektovan.

Ah, ti standardi . . .

Jedan od evropskih standarda je CCITT V.21 standard koji omogućava dvosmernu komunikaciju brzinom od 300 boda, odnosno oko 30 znakova u sekundi. Većina informacionih mreža koristi ovaj standard. Drugi široko korišćeni standard je 1200/75 asimetrični evropski V.23 koji omogućava da se u jednom smeru prenosi 1200 bitova u sekundi, a u drugom samo 75 bitova u sekundi. Ovako radi Prestelov servis jer je uobičajeno da Prestelov računar mora slati stranice informacija, dok korisnik samo sa nekoliko bajtova vrši izbor iz menija. Možda se 75 boda čini jako sporo, ali kada preračunamo da je to sedamdesetak reči u minuti, vidimo da je brzina sasvim razumno odabrana.

Postoji još nekoliko evropskih standarda od kojih pominjemo još 600/75 bod sisteme, a u Jugoslaviji su mreže do sada koriščene za poslovne sisteme i atestirani su modemi jedino za njih sa standardima koji se razlikuju od modema za kućne računare.

Za SAD važe drugačiji standardi. Bell 103 je ekvivalent evropskom V.21, ali sa drugačijim izborom tonova, a Bell 202 odgovara našem 1200/75 s tim što se podaci povratnim kanalom šalju uključenjem/isključenjem tona umesto podešavanjem frekvence.

Donedavno su modemi bili projektovani samo za jedan standard, ali danas u prodaji ima i univerzalnih koji mogu da se prilagode bilo kojem od njih. Iz priložene tabele modema za „spektrum" može se videti da je njihova povećana prilagodljivost praćena i povećanom cenom.

Korišćenje modema

Kako kod nas modemi za kućne računare još nisu zaživeli, iznećemo vam ukratko šta u junskom broju časopisa Your Computer Džek Rasel (Jack Russell) govori o Prestelovim adapterima koji se na britanskom tržištu prodaju za povezivanje računara sinkler—spektrum i ZX81 i ejkorn BBC sa Prestel informacionom mrežom. (U istom broju vlasnici BBC-ja mogu naći i detaljan prikaz Commstara — komunikacionog paketa koji omogućava da se njihov računar koristi bilo kao nemi terminal za

naziv	tip	brzina i dupleks	zahteva li RS232	Cena u funtama (iz 1984)
Concord V22	direktan	1200 pun 600	da	547 300
Dacom Buzzbox	direktan	300/300 pun	da	69.95
Maplin	direktan	300/300 pun	da	39.95 (samo u kitu)
Micro Myte 160 IQ/D	akustični	1032 nestandardni	ne	99
Minor Miracles WS 2000	direktni	300/300 pun 1200/75 pun 1200/75 polu	da	115 (još nije atestiran)
Prism VTX 5000	direktni	1200/75 pun	ne	100, alli cena pala u 1985. na 50
Tandata TM 100	direktni	1200/75 pun	da	99
AS 3/6/12	direktni	univerzalni	da	4190 šilinga

tabela 2 Pregled modema za „spektrum"

neki udaljeni računar, bilo kao inteligentna mašina sposobna da šalje i prima datoteke sa potpunom proverom grešaka što osigurava tačnost prenosa).

Jedan od najomiljenijih tipova modema čija je cena u poslednjih godinu dana pala sa 100 na 50 funti je Prism VTX 5000 za računar spektrum. Korišćenje modema ilustrovaćemo na njegovom primeru.

Kućište u kome se nalazi modem povezujemo sa spektrumom pomoću trakastog kabla sa konektorom direktno u zadnji deo računara. Zahvaljujući konektoru, mogu se priključiti i drugi uređaji kao, na primer, štampač. U telefonsku liniju VTX 5000 se priključuje direktno telefonskim kablom u standardni telefonski priključak na zidu, a sam telefonski aparat povezujemo preko zadnje strane kućišta modema tako da možemo normalno da telefoniramo. Ovaj model, za razliku od većine drugih namenjenih spektrumu, ne zahteva posebnu kupovinu interfejsa RS232.

Čim uključite računar, vidite Micronetov displej, a pritiskom na bilo koju tipku dobijete osnovni meni. Odavde možete kontrolisati kompletan softverski paket namenjen za komunikaciju sa Prestelom i Micronetom koji je upisan u ROM adaptera. Meni nudi: Uključivanje u Prestel mrežu (log on), rad sa terminalom (terminal operation), čuvanje/pregled informacija (save-/view frames), štampanje (print frames), preuzimanje slobodnih programa (download) i mejlboks editor (mailbox editor), kao i povratak u bejzik.

Za pristup Prestelu neophodno je ući u sistem kao sa bilo kojeg erminala na velikom računarskom sistemu. Stoga po izboru odgovarajuće opcije računar traži da unesete svoj ID (identifikacioni broj) koji dobijete upiatom tromesečne pretplate. Po uspostavljanju veze ID se automatski šalje Prestelu i dobijete Prestelov uvodni ekran. Prvo iznenađenje je da vaš spektrum ima displej u 40 kolona. Kad prevaziđete ovaj šok, možete pregledati poruke koje su u međuvremenu za vas prispele, da šaljete sopstvene pripremljene poruke ili istražujete slobodne programe, informacije i sve ostalo što vas interesuje. Stranice sa informacijama možete listati pomoću numeričkih tipki i specijalnih tipki * i £, a povratak na meni ostvaruje se pritiskom na Caps Shift i Enter. Time su omogućene druge funckije paketa kao npr. memorisanje na traci, disketi ili štampanje.

Opcija punjenja nudi vam slobodne programe. Ponuđeni programi nisu neke ekstra klase, ali ako se ipak odlučite za neki od njih, po preuzimanju morate restartovati erminalni program.

Poslednja opcija je editor za pripremu tekstova koji se šalju u mejlboks. On omogućava editovanje samo jedne linije u jednom trenutku ali ne i ispravke u prethodnim linijama. Ako uočite neku staru grešku, moraćete čitavu poruku kucati iz početka.

Raspoložive vrste modema

Na evropskom i američkom tržištu nalazi se mnoštvo modema projektovanih za razne računare i razne standarde. Zadržaćemo se na evropskim tipovima koji se nude za računare spektrum najpopularnije kod nas

Za spektrum se uz Prism VTX 5000 nude još i sledeći modeli.

Concord V22 Data Modem je pravi rols-rojs model. On radi čak i sa izvitoperenom linijom i kada odgovara na poziv sa drugog računara automatski podešava odgovarajuću brzinu.

Dacom Buzzbox ima verzije na struju i na baterije. Verzija na struju je najjeftinija, košta 69.95 funti i ako vam svejedno treba RS232 interfejs to je ekonomičan izbor, ali ne ako vam interfejs nije potreban i za druge namene.

Maplin Modem je najeftiniji na tržištu, ali on je namenjen isključivo onima koji su u stanju da ga sami sklapaju, jer se prodaje jedino u kitu.

MicroMyte 160 IQ/D je akustični kupler koji se može kupiti za 99 funti, ali o je nestandardni model koji umesto interfejsa koristi kasetni ulaz. Na nesreću, ima slabe performanse i fiksiran je na totalno nestandardnu brzinu od 1032 boda. Može vam poslužiti jedino za komunikaciju s drugim računarom koji ima istovetni modem.

Minor Miracles WS2000 World Modem još nije atestiran, ali nudi odlične mogućnosti jer radi i na evropskim (CCITT) i na američkim (Bell) standardima.

Tandata TM 100 kao Prism radi samo na 1200/75 boda u punom dupleksu i koštao je 1984. kao i Prism oko 100 funti. Međutim, za njega je neophodan RS232 interfejs, ali pruža i opciju da uz dodatke smešta 8 telefonskih brojeva od po 16 cifara sa sopstvenom šifrom i log-on kodom. Možete čak promeniti sopstveni ID i šifre.

AS 3/6/12 je jedini modem čiju cenu saopštavamo u šilinzima -4190. Može da radi i po evropskim i po američkim standardima. To je univerzalan modem koji nije namenjen samo spektrumu pa se za svaki računar mora nabaviti i odgovarajuća podrška.

Izbor modema

Pri izboru modema za kupovinu određenog modela treba da budu presudna tri bitna faktora: za šta nameravate da ga koristite, koliko možete da platite i koliko će biti jednostavno njegovo povezivanje sa vašim računarom. Ako vam je modem potreban samo da biste razmenjivali programe sa svojim prijateljima, biće dobar i običan akustični kupler, ali ako želite da pristupate bazama podataka u SAD, onda dobro proverite da li standardi odgovaraju. Ako već imate interfejs RS232 onda su vam, bar teoretski, raspoloživi svi modemi, ali se ipak uverite da će vaš računar raditi sa određenim tipom, jer je kod modema prisutan sličan problem kao kod štampača — računar i periferni uređaj moraju govoriti istim jezikom. Ne zaboravite da je i za rad modema neophodna odgovarajuća programska podrška. Pre svega, treba obezbediti da serijski port radi odgovarajućom brzinom, zatim upućivanje karaktera otkucanih na astaturi na serijski port, kao i dolaznih informacija na ekran. Praktični programi su znatno komplikovaniji, na primer, dobro bi bilo zaustaviti tok podataka sa drugog računara dok vi radite neki drugi posao, recimo otvarate vrata pošto je neko zvonio i slično, da ne govorimo o snimanju informacija na disk ili štampanju poruka. Mada ne isključujemo mogućnost da i sami sastavite programe koji su i bolji od onih na tržištu, ipak preporučujemo da i u ovom slučaju maksimalno koristite gotove programe i stoga je bolje prethodno se dobro raspitati o programskoj podršci. Najzad, većini će biti potreban neko da izvede povezivanje. To nije posao za amatere jer standardne RS232 veze mogu ilekako da variraju, tako da taj neko mora dobro poznavati svoj posao. Imajte sve ovo u vidu ako se odlučite za nabavku modema — veze sa svetom za vaš računar — ali i činjenicu da broj modela iz dana u dan raste, a da cene padaju.

Vladimir Ilijevski
Nevenka Spalević

radanje programskih jezika

Istorija računara

Suma žica na programskom panelu je kasnije zamenjena blokom perforiranih kartica — rupa na određenoj poziciji na kartici imala je značenje prisustva spojne žice na ekvivalentnoj poziciji programskog panela. Jedan blok kartica bi odgovarao sanduku punom programiranih žičanih panela. Montiranje i demontiranje kartica namesto panela za „učitavanje" i izvršenje bilo je mnogo jednostavnije, jeftinije i brže, pa je upotreba retkih i skupih računara postala efikasnija. Za razliku od čvrstih (engl. hard) snopova žica, papirne kartice su bile savitljivive, mekane (engl. soft). pa otuda i naziv mekane stvari ili softver (engl. software) za sadržaj kartica, tj. za programe perforirane na karticama. Pojavom kartica je najedanput postalo jednostavno da se „izbuši" pripremi, program za kompjuter. Za prosto bušenje rupa na karticama diploma iz elektronike više nije bila potrebna. Svako ko bi znao efekat svake rupe sa kartice proizveden na računaru — drugim rečima, oni koji bi naučili tzv. mašinski jezik — mogli bi „bez mnogo muka" da programiraju elektronske računare.

Time je drastično bila proširena baza korisnika računara, sa prethodno privilegovane grupe elektroničara i matematičara, na širok krug stručnjaka gotovo svih profila. Neki su toliko bili opsednuti beskonačnim kombinovanjem perforacija na kartica ma da su čak zanemarili svoje originalno zanimanje i postali. po njihovim rečima. programeri. Mnogobrojne greške i problemi sa rupama na karticama progresivno su se umnožavali sa debljinom paketa kartica. Binarne kombinacije mašinskog jezika se na prvi pogled ponekad gotovo uopšte ne razlikuju. Bilo je vrlo lako napraviti grešku pri programiranju velikih mašinskih programa, a istovremeno vrlo teško naći pogrešni bit u sumi jedinica i nula.

Primitivni jezik mašine

Određene više puta ponavljane kombinacije jedinica i nula (ekvivalent: ima rupe nema rupe) sa određenim efektom na ponašanje računara su počele među mašinskim programerima da dobijaju opisana imena koja bi direktno asocirala na proizvođeni efekat. Često, namesto takvih kombinacija jedinica i nula, programer bi u žurbi da što pre skicira rešenje problema za računar koristio ekvivalentna imena ili skraćenice za ta imena. Jednim spontanim procesom, nastao je programski jezik sloga, sastava, ili ASSEMBLY, kod nas poznat pod pogrešnim imenom *ASEMBLER*. Ako ga već ne zovemo SASTAVNI da se ne bi zamerali Amerikancima i Englezima, bilo bi dobro da ga nadalje zovemo jednostavno *ASEMBLI*. Asembli jezik zahteva od programera znanje „arhitekture" korišćenog računara, tj. sastavnih delova i načina na koji koriščeni računar funkcioniše. Arhitektura računara je elegantan naziv za blok dijagram računara, na kome se vide elementi računara, kao što su mali i brzi memorijski blokovi ili registri sa naznačenim vezama za razmenu jedinica i nula. Da bi naučio *ASEMBLI* jezik, programer bi morao da pažljivo prouči blok dijagram računara i nauči koje su sve moguće razmene bitova između registara. Mnogobrojne mogućnosti su opisane skupom instrukcija (engl. Instruction Set) datog računara.

ASEMBLI jezik koristi podsetne skraćenice ili mnemonike, namesto određenih grupa jedinica i nula iz mašinskog jezika. Ovaj jezik je učinio posao programiranja relativno lakšim jer je bio mnogo jednostavnije, recimo, koristiti skraćenicu M kao instrukciju za množenje ili S za sabiranje namesto binarnih kombinacija 11010101 ili 11011001. *ASEMBLI* programeri su morali da znaju ne samo mnemonike za operacije već i mnemonička imena svih registara. Ime registra bi se koristilo namesto grupe bitova koji bi bili pohranjeni u registru. Tu se javlja fenomen simboličke adrese podatka ili takozvana imenovana variabila.

U potrazi za olakšicama u programiranju, programeri neumorno komplikuju sliku računara. Slično nizovima bitova u mašinskom jeziku, kratki nizovi *ASEMBLI* instrukcija sa čestom primenom počinju da dobijaju mnemonička imena, koja su posle višegodišnje upotrebe polako ušla u sastav repertoara instrukcija *ASEMBLI* jezika. Takvi nizovi smisaonih instrukcija su poznati pod imenom MAKRO (engl. MACRO). Da bi se napravila razlika između direktnog predstavnika arhitekture računara, čistog *ASEMBLI* jezika, nove proširene verzije su nazvane *MARKO SEMBLIJEM*, (engl. MACRO ASSEMBLY). Programiranje u *MAKRO ASEMBLIJU* je bilo mnogo lakše od čistog *ASEMBLI* programiranja. Danas su, zbog svoje neposredne veze sa arhitekturom računara, *ASEMBLI* jezici još uvek neminovni pri sistemskom programiranju i programiranju efikasnih programa.

Ranih pedesetih godina proizvodnja računara je polako počela da postaje unosan posao. Kako su se proizvođači računara umnožavali tako su se sa svakim novim

Ilustracija: Miša Marković

Novi saradnik

Naš novi stručni saradnik i dopisnik iz Njujorka, Radomir A. Mihajlović-Bili, vanredni je profesor za predmet Operativni sistemi na njujorškom Institutu za tehnologiju (New York Institute of Technology). Radomir Mihajlović završio je studije elektronike na beogradskom Elektrotehničkom fakultetu, magistrirao je elektroniku i matematiku i odbranio doktorat iz oblasti Statističke teorije komunikacija i Teorije informacija. Kaže da je, počevši intenzivno da se bavi kompjuterima, naučio mnogo o ljudima i da je za njega Computer Science, zapravo, nauka o ljudima, a ne o računarima. Naši čitaoci imaće priliku da bolje upoznaju Bilija Mihajlovića, profesionalnog programera koji tvrdi da su pirati najprogresivnija snaga naše informatičke revolucije, kroz intervju koji smo pripremili za sledeći broj „Računara".

Nekada davno, u vreme samog početka ere elektronskih računara, programi su bili sastavljeni od mreže žica. Dve mogućnosti, prisustvo i otsustvo spojne žice, su bile, radi jednostavnosti, označene jedinicom i nulom. Kako je reč o samo dve mogućnosti, koje je lako izbrojati na prste, elektronski računari su nazvani binarnim digitalnim računarima, što treba da zahvali latinskim rečima ili dva i digit ili prst. Paneli sa programom su se, po potrebi, montirali na šasiju računara, pa otuda arhaična ideja da računarima mogu da rukuju samo stručnjaci za elektroniku. Jednostavna instrukcija tipa „saberi 2+3" zahtevala je višečasovno povezivanje, repovanje, programa. Ako bi, kojim slučajem, rezultat bio 4, programer je morao da pretraži čitav žbun finih žica u potrazi za „bubicom" (engl. bugg), tj. pogrešnom vezom. Ponekad ni dani pretraživanja, (engl. debugging), ne bi bili dovoljni. Tako je u muci rođeno zanimanje koje se danas nosi možda s najviše dostojanstva — zanimanje zvano programer.

tipom računara programeri ponovo morali da late učenja novog *ASEMBLI* jezika. To i jeste i nije bio problem za radoznale programere. Međutim, ono što je bilo važnije je da su svi programi napisani tokom dugog perioda programiranja na starom računaru morali da budu izpočetka ponovo pisani. Taj problem među kompjuterašima je poznat kao problem prenosivosti (engl. Transportability) softvera. Programi sa jednog tipa računara ne važe za drugi tip računara — znači da dati programi nisu prenosivi.

Potreba za daljim pojednostavljavljenjem pripreme programa, iskustva sa *MAKRO ASEMBLIJEM* i zahtev za programima koje ne bi trebalo svaki čas prevoditi sa jezika jedne mašine na jezik druge, doveo je do pojave jezika na visokom nivou, univerzalnih jezika razumljivih od strane računara različitih tipova. U jezicima visokog nivoa mnemonici iz *ASEMBLIJAi MAKRO ASEMBLIJA* su zamenjeni kompletnim rečima i čoveku lako razumljivim simbolima. Poseban program bi generisao odgovarajuće blokove jedinica i nula u mašinskom jeziku na svaku instrukciju u visoko-ninovskom jeziku visokog nivoa. Instrukcije u jeziku visokog nivao bi bile ulazni podaci, a jedinice i nule mašinskog koda bi bile proizvedeni izlazni podaci program-prevodioca. Takav program bi imao funkciju čitača tabele za prevođenje sa čoveku razumljivog, izvornog (engl: Source Code), programa na mašinski program, (engl. Object Code). Programi za prevod jezika visokog nivoa su poznati danas kao prevodioci ili kompajleri (engl. Compiler). Prisutvo kompajlera je neophodno u računaru za rad sa odgovarajućim jezikom visokog nivoa.

Bejzik ili Pomrsik?

Matematičari su uredili sebi pogodan jezik za prevod matematičkih formula, FORmula PREVodioc (Engl. FORmula TRANslator) na mašinski jezik, poznat danas u čitavom svetu kao FORTRAN. Po sličnoj logici. *FORTRAN* bi se mogao kod nas jednostavno nazvati FORPREV(?P) *FORTRAN* se pokazao kao vrlo pogodan jezik za programiranje rešenja problema sa malim brojem podataka i velikim brojem matematičkih operacija sa podacima. Između preko hiljadu programskih jezika predloženih sa različitih strana, zanimljivo, *FORTRAN* se proslavio u velikom delu zbog toga što ga je IBM, najveći svetski proizvođač računara. usvojio za službeni jezik. Mnogi korisnici su, svidalo im se to ili ne, morali da koriste *FORTRAN* i mnogobrojne škole su morale da uče svoje studente *FORTRANU*. Za razliku od *ASEMBLIja* i mašinskog jezika. isti program napisan u *FORTRANU* bio je funkcionalan, pred IBM računara. na bilo kom drugom tipu računara koji bi imao sebi svojstveni *FORTRAN* kompajler. *FORTRAN* programi su postali transportabilni. dok su jedino kompajler programi ostali neprenosivi između mašina različitih tipova.

Slično slučaju FORTRAN-IBM, Američko ministarstvo za obranu (engl: Department Of defence ili DOD) je zaključilo da za potrebe svoje administracije, koja je verovatno najveća na svetu, mora da ima jezik pogodan za programiranje rešenja za obradu ogromnih količina podataka koristeći mali broj operacija na njima. Predložen je Jednostavni Poslovno Orjentisani Jezik, (engl. COmmon Business Oriented Language ili COBOL), skraćeno kod nas poznat kao KOBOL. Naš ekvivalent JEPOJ zvuči muzikalno.

Obadva jezika. *FORTRAN* i *KOBOL*. zbog svog relativno dugog veka, danas su najviše „govoreni" jezici na svetu među programerima. između ostalog i zahvaljujući IBM-u, računarskoj, i DOD-u, vojnoj supersili, koje su ih promovisale.

ALGOL je bio jezik koji je u više verzija uglavnom imao za cilj da postane univerzalni programski jezik, sa svim mogućim instrukcijama i tipovima podataka, pogodan za masovnu obradu podataka, nešto slično *FORTRANU* i *KOBOLU* zajedno. Kao vrlo glomazan, među programerima, ne baš puno ornim da uče više nego što je minimalno potrebno, *ALGOL* nikada nije mogao da postane široko prihvaćen. Mada obožavan od teoretičara programiranja, nije mogao da osvoji širi krug svakodnevnih, običnih softveraša. Međutim, i pored negativnog odnosa programera prema *ALGOLU*, ovaj je jezik verovatno odigrao jednu od najznačajnijih uloga među dosadašnjim jezicima. Sa svojim mnogobrojnim mogućnostima pune je uticao na konačni oblik mnogih kasnije predloženih programskih jezika. Tipični primeri su: *PASKAL*, C, *SIMULA*, PL/I, *BLISS*, *ADA* itd.

Mašine koje je IBM prodavao ili rentirao i koje je DOD koristio bile su velike sa ogromnim memorijskim kapacitetima. Zbog memorijskog luksuza u kome su programirani prevodioci *FORTRAN* ili *KOBOL* tekstova na tekstove u mašinskom jeziku sa samo dva simbola, prevodioci ovih jezika su napravljeni tako glomaznim da ih je bilo gotovo nemoguće koristiti na mirko računarima sa siromašnom memorijom. Kada su se ranih sedamdesetih godina na tržištu pojavili mikroračunari, sa malim i skupim memorijama i sa svojom originalnom namenom da služe prvenstveno za zabavu, obrazovanje i rešavanje jednostavnih problema obrade malih količina podataka, novi programski jezici su počeli da se javljaju kao pečurke posle kiše. Činilo se da skoro svaki obdarjeni programer ima svoj jedinstveni programski jezik. Jedan od prvih takvih jezika, pa zbog toga i najpoznatiji među hekerima, je Početnički Mnogonamenski Simbolički Instrukcioni Kod, (engl. Beginner's Allpurpose Symbolic Instruction Code ili BASIC). Da smo ga kojim slučajem prvi predožili, zvao bi se najverovatnije PMSIK ili PoMrSiK a ne BEJZIK, što se odomaćilo među našim programerima. Ovo je bio jezik koji je bilo moguće brzo i lako naučiti i to mu je verovatno i jedina dobra osobina. U osnovnim školama, gde problemi i rešenje za programiranje nisu komplikovani, *BEJZIK* IZGLEDA KAO BLAGOSLOV — LAKO GA JE NAUČITI I JOŠ LAKŠE KORISTITI. *BASICna* slika je daleko tužnija u slučaju ozbiljnih programskih projekata. vrlo je teško naći „bug" u velikom *BASIC* programu pa izgleda da mu naše ekvivalentno ime POMrSik ili BIEJeZIK odlično pristaje. Kada popravku programa ne vrši autor već korisnik, svaka nova strana *BEJZIK* listanja (eng. listing) postaje mučno i traumatično iskustvo. Osobine *BEJZIKa* su uslovljene karakterističnim ograničenjima malih računara — mali kapacitet memorije i skromne procesorske mogućnosti, pa stoga i laki problemi i kratki programi. Bilo kako bilo, zahvaljujući milionskoj invaziji mikroračunara, *BEJZIK* se proslavio. Na opštu žalost oboževaoca *BEJZIKa*, sa padom cena elektronskih komponenti i povećanjem računarskih moći mikrića *BEJZIK* sve više potiskuju skraćene verzije giganta kao što su *FORTRAN* ili *ALGOL*.

Paskal bez Bleza

Ne sa ciljem da udovolji ograničenjima hardvera kao *BEJZIK*, već da udovolji programerima i olakša im pisanje velikih programa, računarski genije iz Ciriha Niklaus Wirt projektuje 1971 god. danas već široko prihvaćeni jezik *PASKAL*. Za razliku od dosada pomenutih; *FORTRANA*, *KOBOLA* i *BEJZIKA*, direktan prevod *PASKAL* na naš jezik je opet PASKAL. *PASKAL* je ime poznatog matematičara i fizičara. Moguće je da je baš to razlog što su ga profesori mnogih univerziteta vrlo brzo prihvatili. Wirt je prethodno puno radio na razvoju jezika sa profesionalnom namenom kao što su Programski Jezik br. 1, (Engl. Programming Language 1 ili PL/1), i *ALGOL*. Njegova iskustva sa ovim jezicima su, u svakom slučaju bila presudna po sudbinu i izgled *PASKALA*. *PASKAL* se danas svrstava u grupu jezika tipa

ALGOL ili u grupu strogo tipiziranih jezika čiji je lider *ALGOL*. Svi programski jezici visokog nivoa koriste imena varijabila kao oznake nizova jedinica i nula specificirane dužine. Specifičnije dimenzija i unutrašnje podele i organizacije ovih nizova je postalo poznato kao specifičiranje tipa podataka. Strogo tipizirani jezici kao što je *Paskal* izričito zahtevaju deklarisanje svih varijabila na samom početku programa.

Specificiranje tipa podataka i odgovarajućih mogućih operacija sa datim tipom podatka dovelo je do pojave astrakcije podataka. Wirtovo profesorsko zanimanje je očigledno uticalo na *PASKAL*. Glavna ideja autora *PASKALA* je bila da u slučaju složenog problema rastereti programera-studenta od nevažnih šifrovanih detalja oko pisanja instrukcija, već da akcenat stavi na samo rešavanje problema. Na skoro svim univerzitetima u SAD, sa retkim izuzecima, *PASKAL* je obavezni programski jezik.

PASKAL je jezik na visokom nivou u pravom smislu te reči, pogodan kako za male tako i za velike računare, sa ključnim karakteristikama jezika visokog nivoa. Prema nekoj vrsti definicije, jezik na visokom nivou bi trebalo da zadovolji sledeće uslove:

(1) Jezik ne zahteva od programera znanje arhitekture mašine i mašinskog jezika.

(2) Jezik je nezavistan od posebnih karakteristika datog računara.

(3) Postoji jedan prema više relacija između originalnog, izvornog, programa i mnogobrojnih objekat programa u mašinskom kodu različitih mašina.

(4) Oznake i instrukcije jezika su prirodne i očigledno razumljive sa slobodom pisanja bez tabelarnih pravila pisanja i ograničenja.

Programiranje u *PASKALU* počinje sa definicijom problema i tipova podataka, zatim sledi podela problema na manje parcijalne probleme sa početkom i završetkom (engl. begin i end) koji se dalje razbijaju na elementarne blokove takođe sa strikt no naznačenim početkom i završetkom. Sa *PASKALOM* se javlja posebna disciplina organizovanog i lepog pisanja programa, tzv. struktrano programiranje, (engl. structured programming).

Mnogostruki i jedinac

Pronalazak program-prevodioca i jezika visokog nivoa doveo je do masovne revolucije u programiranju. Jednom napisani programi su postali univerzalno primenljivi bez ili sa vrlo malo izmena. Produktivnost u poslu programiranja postaje značajan faktor. Javlja se tzv. discpilina softver inženjerstva (engl. Software Engineering). Nešto što je nekada bilo hobi ili neizbežna rutina u eksperimentisanju sa brojevima postaje punopravno zanimanje. Međutim, kao što to uvek biva sa svim ljudima, uvek nezadovoljni datim stanjem stvari, neskromni i ambiciozni softver inženjeri nisu zadovoljni prenosivošću programa visokog nivoa, već zahtevaju prenosivost i univerzalnu primenljivost i program-prevodioca. Sa tim zahtevima dvojica zanesenjaka iz Belove Laboratorije, Karnigan i Rici (Brian Kernighan i Dennis Ritchie), pređlažu vrlo prenosiv kompajler za jezik koji oni iz šale nazivaju jezik C. Za razliku od istraživača sa poznatog M.I.T.a koji su radili na programu MULTIX, u prevodu MNOGOSTRUKI, koji bi mogao da se po potrebi ponaša kao mnogo različitih računara, Karnigan i Rici predlažu program UNIX ili JEDINAC, koji bi mogao da se montira lako na bilo koji računar i koji bi drastično olakšao pripremu novih programa. Pri pisanju UNIXA silom prilika, da bi sebi olakšali posao, u zavisnosti od problema sa kojima su se pri tom sretali,

Augusta Ada: Prvi ženski programer na svetu

Karnigan i Rici su morali da izmišljaju nove instrukcije i pomoćne male programe, da bi na kraju složili ove u programski jezik B. Naziv B im se iz nekog razloga verovatno nije mnogo sviđao, pa su jezik prekrstili u jezik C. Programski jezik C je danas poznat kao jezik sa najlakše prenosivim kompajlerom, pa stoga i sa najlakše prenosivim programima. Da bi bio ono što jeste, tj. vrlo prenosiv, jezik C je koncipiran kao vrlo mali jezik. Bez mnogo različitih instrukcija je teško izvrnuti bilo šta u jeziku da bi navodno bilo bolje nego u originalnom jeziku. To znači da sem originalne verzije, gotovo uopšte nema dijalekta jezika C.

Godine 1974, suočen sa astronomskim porastom troškova razvoja softvera pisanog upotrebom mnogobrojnih programskih jezika (jedna studija navodi da je u jednom momentu bilo u upotrebi oko 5000 različitih programskih jezika i dijalekata), američki DOD odlučuje da formira službeni programski jezik. Kako nijedan postojeći jezik nije zadovoljavao, DOD je delegirao nekoliko različitih grupa sa zadatkom da predlože svoje verzije „super jezika". Posle dužeg procesa modifikovanja i integracije različitih predloga, uspostavljen je programski jezik ADA. nazvan tako u čast Auguste Ade ćerke lorda Bajrona i asistentkinje Čarlsa Bebidža, pronalazača prvog praktično upotrebljivog računara. Zbog toga što će biti obavezni jezik programera koji budu radili za najvećeg svetskog naručioca softverskih poslova, očekuje se da će ADA biti jedan od najvažnijih jezika osamdesetih godina.

ADA je ogroman jezik i nije čudo da ima mnogo osobina koje drugi popularni jezici nemaju. Jezik je projektovan sa prvenstvenim ciljem lake čitljivosti, stroge tipiziranosti, mogućnosti paralelnog programiranja više instrukcija istovremeno (engl. concurrent programming) i masovne prenosivosti između mnogobrojnih tipova računara danas u upotrebi u oružanim snagama SAD. Kako bi se obezbedilo da skoro sve moderne osobine budu uzete u obzir, ideje i sugestije za jezik su bile prihvatane od programera sa svih strana sveta. Mnogi programeri, po onoj našoj „gde ima puno kuvarica čorba je preslana", smatraju da je *ADA* preglomazan i ne baš pogodan jezik za upotrebu. Međutim, činjenica da Sovjeti užurbano prevode sve knjige o jeziku *ADA* i da proizvođači personalnih računara skraćuju jezik na podskup i nude različite verzije obećava izvesnu budućnost.

Kako se svetu ne može ugoditi, istorijski, jedan programski jezik nije mogao nikada da zadovolji svakoga. Najbolji programski jezik je uvek bio tu iza ugla, čekajući da bude pronađen. Postavlja se pitanje da li je moguće naći konačni programski jezik i kako će on izgledati? Kakav će izgled imati popularni jezik jednom kada *ASEMBLI BEJZIK* i *PASKAL* postanu eksponati u muzeju kompjuterskih nauka? Zanimljivo, izgleda da je na ova pitanja jednostavno odgovoriti: biće više konačnih jezika u upotrebi. Nimalo čudno, mi već znamo imena tih jezika. Biće to jezici poznati kao engleski, srpskohrvatski, francuski, japanski itd. Jer, bez diskusije, najlakši jezik je prirodan ljudski jezik. Na žalost, to ostaje kao problem u nasleđe budućim programerima. Da bi razumeo prirodan jezik, računar mora biti predhodno programiran programima veštačke inteligencije, koji bi se s lakoćom hvatali u koštac sa dvosmislenostima, alegorijama, hiperbolama, smicalicama, štosovima i ostalim nejasnoćama poetičnih programera.

Računar budućnosti mora biti u stanju da razluči bitno od nebitnog — da odlučuje u pravom smislu te reči. Na današnjem nivou razvoja softver inženjerstva postoji grupa od nekoliko jezika sa prvenstvenom namernom upošćavanja problema informisanja računara o tome šta je očigledno i bitno, a šta nije ili šta se implicitno podrazumeva. Tipični pretstavnicu su: U SAD popularni *LIPS* sa MIFA. u Japanu glavni jezik računara pete generacije. *PROLOG*. Zbog elektronske supersile iza sebe, *PROLOG* verovatno ima bolje izglede za uspeh. Mada ime PROLOG dolazi od skraćenice PROgramiranje.sa LOGikom (engl. PROgramming in LOGic), nimalo nije tačno da je potrebno da programer bude ekspert za matematičku logiku da bi koristio *PROLOG*. Naprotiv, jezik je toliko jednostavan da ga i deca mogu lako naučiti. *PROLOG* je jezik pogodan za obradu podataka znanja. Program u *PROLOGU* se sastoji od činjenica o određenom subjektu. Moguće je postaviti pitanje *PROLOG* programu i ovaj će pokušati da odgovori, koristeći prethodno programirane činjenice. Ograničenja *PROLOGA* ili *LISPA* su ogromna. Međutim, u svakom slučaju, prvi koraci ka programskim jezicima budućnosti su učinjeni.

U današnjem trenutku razvoja kompjuterskih nauka i softver-inženjerstva, posle realativno duge istorije sa mnogobrojnim jezicima, pomalo izgleda smešno stav autora *FORTRANA* IZ 1954. god. da će *FORTRAN* svojim savršenstvom i razumljivošću omogućiti jednostavno programiranje svih algoritama i gotovo u potpunosti eliminisati pogrešno kodovanje i potrebu za pretraživanjem i popravkom programa.

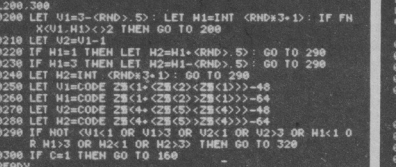
Dr Radomir A. Mihajlović

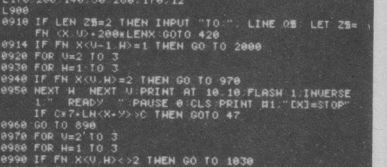
Ekskluzivno

ekranski editor i druge bajke

Naša nova trakoteka

Svi se sa dosta nostalgije sećamo dana kada je časopis „Galaksija" imao Biblioteku programa za džepne programabilne računare. Ta biblioteka je delovala potpuno amaterski: naši čitaoci su slali programe, mi smo birali najbolje i objavljivali njihove naslove. Dok su autori primljenih programa dobijali satisfakciju jedino kroz objavljivanje imena i izbor dva besplatna programa iz Biblioteke, drugi su čitaoci, plaćajući troškove koji su pokrivali ekonomsku cenu fotokopiranja i poštarine, mogli da naruče sve programe koji ih interesuju i dobiju ih u nekom razumnom roku. Pre skoro dve godine smo pokušali da uradimo nešto slično, pa smo osnovali Biblioteku programa za kućne računare, ali se brzo pokazalo da njeni principi izgledaju arhaično u naglom talasu komercijalizacije i — stvar je pala u vodu. Sada pokušavamo ponovo, ali, sasvim u duhu vremena, na nešto komercijalnijoj osnovi.





Čovek se, kaže, uči na greškama: pre svega, distribuirajući programe za džepne računare nismo imali nikakvu konkurenciju pa je pojava svakog dobrog programa bila pravi mali praznik. Kada je reč o „spektrumu", „komodoru" i „amstradu" i drugim kućnim ljubimcima, naša konkurencija je vrlo moćna: čini je armija **pirata.** Kada se za 15 dinara može nabaviti program koji u Engleskoj košta otprilike isto toliko funti, mali programi iz naše Biblioteke će interesovati samo mali broj čitalaca. S druge strane, prošla su vremena kada su džepni računari bili isključivo hobi: neko ko uloži mnogo rada u izbiljan program obično želi da za to dobije i odredenu materijalnu satisfakciju.

Biblioteka na kaseti

Odustajanju od Biblioteke programa za kućne računare kumovale su, da budemo iskreni, i izvesne organizacione teškoće: ako „na lageru" imamo dosta programa, onome ko ih snima je teško da odredi koje treba da smesti na koju kasetu i kojim redom, onome ko kopira uputstva koja uputstva da kopira u koliko primeraka i, najzad, onome ko prikuplja materijal i šalje pisma da sinhronizuje te dve delatnosti. Ako nam ne verujete da je sve ovo teško organizovati, pogledajte malo preko granice: **ni jedan jedini** strani časopis nema ništa slično Biblioteci programa! Daleko je jednostavnije i jeftinije izdavati po jedan ili dva programa na svakoj kaseti, umnožiti ove kasete, štampati uputstva i svima zainteresovanima slati jednake pakete.

Ovaj prilično dugačak uvod je imao za cilj da vam objasni razloge koji su nas naveli da se upustimo u izdavanje softvera i da vas (a pomalo i autora ovoga teksta) uveri da ne odustajemo tako lako od amaterskih principa, koji nam se i dalje dopadaju. Časopis „Galaksija" je, dakle, odlučio da počne da izdaje kompjuterske programe na kasetama i otuda ovaj tekst, kao i narudžbenica koju ste odmah primetili u dnu strane.

Kakve programe želimo da izdajemo? Naravno, dobre programe! A u takve ubrajamo, pre svega, ozbiljne sistemske programe, poslovne aplikacije i, ako ih bude bilo, igre koje kvalitetom konkurišu stranima. Softver koji budemo izdavali će biti potpuno originalan i moći će da se nabavi isključivo posredstvom našeg časopisa (i posredstvom pirata, ali o tome docnije).

Podstaknuti dovitljivošću jednog našeg čitaoca, planiramo da na kasetama objavljujemo još jednu vrstu programa. U znak zahvalnosti potpuno besplatno ponavljamo njegov mali oglas:

Prodajem program Devpak v7.8. To je poboljšana verzija dobro poznatog paketa Mons-Gens 8M obogaćena sa oko 15 korisnih rutina. Cena sa kompletnim upustvom za rad je 600 dinara. Gabor Erdeg, Vareška 26, 24000 Subotica.

Možda ste već shvatili o čemu se radi: naš čitalac iz Subotice prodaje unapređenje programa DEVPAC koje je u „računarima 6" objavio Vlada Kostić. Objavljeno je par strana listinga koga, priznajemo, nije lako otkucati. Gabor Erdeg je obavio taj posao i sada olakšava život ostalim čitaocima „Ra-

Kako sam pisao editor

Na ideju da napišem ekranski editor došao sam sasvim slučajno. Dobio sam program BETA BASIC V1.8 u čijem uputstvu za upotrebu piše: „Korisnici naredbe SCREEN i KEYIN, svako željan izazova može da napiše pravi ekranski editor na bejziku!". Kao svaki haker koji drži do sebe, zaključao sam vrata svoje sobe, uključio mašinu i kada je svanuo dan, ja sam imao gotov ekranski editor napisan na bejziku. Bilo je to, naravno, pusto hakersko zezanje, ali me je navelo na neka razmišljanja.

Pre svega, „spektrumov" bejzik je sasvim dobar. Svaka čast programerima koji su pisali BETA BASIC, HIPER BASIC, MEGA BASIC i slične programe, ali za tim nema bog zna kakve potrebe. Lepo je imati, recimo, sprajtove na bejziku, ali čemu to kada je bejzik i tako katastrofalno spor za akcione igre? Uz to, program napisan na, recimo, BETA bejziku neće moći da radi bez BETA bejzika, što predstavlja dodatnih 10 K memorija. Takav program se ne može prodati, objaviti u nekom časopisu ili, jednostavno, dati nekome ko nema, ili ne zna da koristi BETA BASIC.

S druge strane, „spektrumov" editor je ispod svake kritike. Toliko je loš da prosto destimuliše korisnika, više nego same gumice. Da ne spominjemo flešinarje ekrana dok se kursor pomera gore (dole ili genijalnu proceduru LIST/BREAK/CAPS) 1, da bi se editovala neka linija. Sećam se da sam svojevremeno koristio BETA BASIC samo zbog EDIT naredbe.

Imajući sve to u vidu, nije mi bilo teško da zaključim da je „spektrumu" potreban ne neki novi ultra bejzik, već dobar pravi ekranski editor. Sledeći korak bio je pisanje ekranskog editora na mašinskom jeziku. Osnovna ideja je bila: dovedi kursor na mesto greške, ispravi, pritisni enter. Još nešto: editor mora da bude prijatan za rad. Džaba sve mogućnosti ako se korisnik oseća frustriranim.

Prvi problem koji je trebalo rešiti bila su 32 znaka u redu. To je i suviše malo da bi bejzik program bio čitljiv. Taj broj se može povećati na 42 (matrica 5×7), 51 (4×6) ili 64 (3×6). Prvo rešenje i nije neko poboljšanje, a 64 je suviše zbijeno (jeste li ikada koristili TASWORD II?), tako da sam se opredelio za 51 karakter u redu. Za divno čudo, slova su ispala izuzetno lepa i jasna. Negde februara ove godine, počeo sam da pišem rutinu za prikazivanje pedeset i jednog znaka u redu koja je, zajedno sa novim karakter setom i svim pripadajućim potprogramima (za brisanje, pomeranje ekrana gore, dole, itd) zauzela 2,5 K memorije. Tako je počeo ceo posao oko pisanja ovog programa.

Vertikalni razmak između linija iznosi tri tačkice — taman da se kursor u obliku crtice provuče između — tako da se na

Editor, kao opsesija: Vladimir Kostić u redakciji „Galaksije"

čunara"! Samo, znate, **mi** smo imali taj program pre svih, pa i pre davaoca ovog oglasa. Mi smo, dakle, mogli da ponudimo svima koji su spremni da se odreknu nekih 400 dinara da bi izbegli nekoliko časova kucanja u „spektrumovu" tastaturu ovo unapređenje Devpaca na kaseti! Tako bi određeni procenat od prihoda pripao Vladi Kostiću, koji ga je, priznajte, ipak malo više zaslužio od Gabora Erdega. Na kasetama ćemo, dakle, izdavati i duže programe koje objavljujemo u „Računarima", pri čemu ćemo se potruditi da cena ovakvog softvera bude minimalna — kasete ovoga tipa predstavljaju pomoć čitaocima, a ne način da dopunimo svoj budžet! Praktikovaćemo i objavljivanje opisa programa i sve potrebne dokumentacije u „Računarima", dok ćemo umesto dugačkih listinga nuditi kasete, oslobađajući prostor u časopisu za druge tekstove.

Po narudžbenici vidite da za početak nudimo pet programa: 'Ekranski editor' Vlade Kostića, 'Hiper bejzik' Igora Fischera, 'Velika akcija' Ace Radovanovića, 'Eatenglish 1' Slavoljuba Milekića i Dragana Tanaskovskog i 'Trodimenzionalna grafika' Jovana Skuljana. 'Hiper bejzik', 'Veliku akciju' i 'Eatenglish' ne treba posebno opisivati — radi se o programima koji su dobili prvu, drugu i trecu nagradu na našem prošlogodišnjem Konkursu i o kojima smo dosta pisali kako u „Galaksiji" tako i u „Računarima". 'Trodimenzionalna grafika' je program koji prati tekst u ovom broju našeg časopisa i koji je omogućio crtanje svih efektnih slika koje smo objavili i mnogih drugih za koje nije bilo mesta. Ostatak ovoga teksta posvećujemo 'Ekranskom editoru', programu koji nas je, da budemo iskreni, i naveo da počnemo da razmišljamo o izdavanju kaseta i kao takav zaslužen poneo broj 1 u našem spisku.

Ekranski editor

Poznajete li Vladu Kostića? To je saradnik „Računara" trenutno zadužen za vrhunske programe za „spektrum" i „cake" u vezi sa mašinskim programiranjem na Z80. Vlada je (nemojte mu reći da smo ovo napisali, jer se ljuti kada ga neko hvali) vanserijski programer koji će, kada naiđe na dobru ideju, za samo jednu noć napisati nešto kao što je Sintaksni teror, 256 boja na Spectrumu ili, ako mu date nekoliko noći više, unapređenje Devpaca. I pored takve efikasnosti, na sva pitanja tipa „Šta radiš" Vlada Kostić je poslednjih devet meseci odgovarao „Pišem ekranski editor". Veoma nas je obradovalo kada je ovaj odgovor nedavno prerastao u „Završavam ekranski editor". To je, ujedno, bilo i zeleno svetlo za ovaj napis.

Program na koji jedan vrhunski programer potrošio devet meseci bi trebao da bude neka dobra stvar. Verujemo da je ekranski editor baš to: dobar i koristan program kome na domaćem (a, verujte, ni na stranom) tržištu nema konkurencije! 'Ekranski editor' je program tipa „beta bejzika": olakšava rad i život pri pisanju drugog softvera. Videćete i kako.

Čak se i najzagriženiji ljubitelji „spektruma" slažu sa konstatacijom da je editor ovoga računara očajno slab: naredbe se kucaju isključivo traženjem odgovarajuće reči na tastaturi, da biste ispravili neku liniju morate da otkucate LIST nn, zatim da pritisnete ENTER pa da pritisnete SHIFT i 1 (EDIT) i da onda polako šetate kurzor i mučite se sa izbacivanjem i umetanjem slova, ukoliko editujete deset sekcesivnih linija, morate deset puta da kucate EDIT!

Kod ekranskog editora stvar je daleko jednostavnija: otkucate L (ili LIST) i broj linije i na ekranu će se pojaviti ta linija kao i (ako ste želeli) segment programa iza nje. Koristeći četiri strelice pokretaćete kurzor po ekranu i, na mestima na kojima želite, ubacujete, brišete ili menjate tekst. pri čemu naredbe možete da kucate slovo po slovo kao na svim drugim „normalnim" računarima. Pošto ste u liniju uneli ispravke, pritisnućete ENTER i ona će biti unesena u program; pritiskajući donju strelicu možete za trenutak da predete na ispravljanje sledeće.

Obzirom da rad sa 32 slova u redu teško može da bude dovoljan za fleksibilnost ekranskog editora, znakovi su malo suženi, pa ih u svaki red stane po 51. Bez obzira na to, matrica 4*6 je dovoljno velika da slova budu sasvim čitljiva čak i na televizorima slabijeg kvaliteta; jedino se neka mala slova (npr. k) teško razlikuju od velikih. Dizajn ekrana je unapređen i tako da su linijski brojevi vizuelno odvojeni od korisnog dela naredbe, što znači da su problemi prepoznavanja početka i kraja linije i njenog broja koji se meša sa adresama u raznim POKE i USR naredbama stvar prošlosti.

Osim „običnog" ispravljanja linija, ekranski editor nudi i mnoge druge opcije koje će biti dragocene pri razvoju programa: mogućnost brisanja, prebacivanja i kopiranja većih programskih segmenata, prenumeracija (naravno uz vođenje računa o GOTO, GOSUB i sličnim naredbama), povezivanje većeg broja linija u jednu i razdvajanje jedne linije u njih nekoliko, dampovanje sadržaja ekrana na nekom štampaču (ne na Sinclairovom, jer on jedva može da podnese 32 slova u redu) i, da li ste to očekivali, rad sa funkcijskim tasteri-

ekranu vidi 20 linija. A na samom dnu je statusna linija koja zaslužuje posebnu pažnju. U njoj se stalno prikazuje „oblik kursora" (L,C ili G), pozicija kursora, način rada editora (NORMAL, AUTO, EDIT, FIND, LIST ili LLIST), da li su dve gornje linije zamrznute, dužina bejzik programa i da li je uključen EDITOR ili INSERT MOD.

Vrlo tezak (filozofski) problem bio je kako će se program kucati: jedan taster — jedna naredba ili slovo po slovo. Odmah da kažem da su, po mom saznanju, ZX81 i „spektrum" jedini kompjuteri na svetu koji koriste tokenizovanu tastaturu. Za to postoji odlican razlog — loša tastatura — ali ostaje da je to vrlo komplikovano i mučno. Šift prvi, šift drugi, šift treći . . .

Postoji još jedan problem vezan za sam ekranski editor: kada se na „spektrumu" edituje linija, kursor može da se šeta samo levo/desno, pa računar vrlo lako prati u kom se modu nalazi. Kod ekranskog editora je kursor moguće dovesti bilo gde na ekranu. Kako sada računar da zna da li sledi „keyword" ili obično slovo? Mogao bi da analizira ono što se nalazi ispred kursora, ali to je isuviše komplikovano. Opredelio sam se za kucanje slovo po slovo, i mislim da nisam pogrešio. Da bi se korisnicima olakšao rad, moguće je koristiti skraćenice: P. umesto print, na primer. A postoji još i kursor za kopiranje (koji se od glavnog razlikuje time što se ne fliše), pa ako nešto što treba kucati već postoji na ekranu, jednostavno se iskopira. Naravno, ni

funkcijski tasteri nisu zaboravljeni.

Pošto je koncepcija bila tu, krenuo sam sa izradom. Gruba blok šema glavne izvršne petlje zauzela je dobar deo zida u mojoj sobi, i po njoj je program polako počeo da dobija svoj konačni izgled. Ta faza je jako dugo trajala — dobrih 6 meseci. Šest meseci izuzetno napornog rada (preko 400 stranica napisanog programa), ali se sigurno isplatilo. Dok ovo pišem, editor već uveliko radi. Ostalo je da se uvedu još neke naredbe, i naravno, da se ispravi poduži spisak sitnih nepravilnosti u radu. Koliko će to trajati, ne usuđujem se da prognoziram, mada se potajno nadam ne više od 30 dana.

Šta korisnik može da očekuje od ovog editora? Pre svega, jedan izuzetno profesionalno urađen program po najvišim svetskim standardima i, naravno, jedno korisno pomagalo koje će mu omogućiti da brzo i lako piše i ispravlja bejzik programe, sa jednim zadovoljstvom do sada poznatim samo na daleko skupljim kompjuterima — često preko 1000 funti. Cena koju za to treba platiti je 16 K memorije i dva minuta nerviranja pri učitavanju, ali sve na ovome svetu ima svoju cenu, pa i jedan ekranski editor.

Možda je red da na kraju jednog ovakvog teksta iznesem i neka svoja privatna zapažanja. Pre svega, mislim da sam i sebi i drugima dokazao da za pisanje ozbiljnih profesionalnih programa nije potrebno ništa posebno profesionalna oprema. Na početku sam koristio najobičniju „spektrum"

doduše sa profi tastaturom, i kasetofon. Kasnije, kada je program počeo da dobija na dužini, opremu sam dopunio sa dva mikrodrajva i 80 K memorije. U našim uslovima, vrednost takvog sistema iznosi petnaestak starih miliona što priznaćete, nije neki ogroman izdatak. Druga, vrlo zanimljiva stvar je da za pisanje programa 16 K nije potrebno samo šesnaest puta više vremena nego za 1 K dugačak program; da je tako, trebalo je da celu stvar završim krajem februara. Sigurno da kao iskusan programer nisam očekivao tako kratak rok, ali nisam ni u snu sanjao devet meseci! Možda delimičan odgovor leži u tome da većina standardnih rutina iz „spektrumovog" roma (za ispisivanje po ekranu, očitavanje tastature, proveru sintakse, itd) nije bila upotrebljiva za ovaj editor. Uz to, trebalo je napraviti ceo jedan razvojni sistem koji će omogućiti pisanje tolikog programa na jednom skromnom „spektrumu" (samo sors fajlovi zauzimaju 140 K memorije).

Mnogi me pitaju zašto baš pišem editor, a ne neku igru. Iskustvo mi pokazuje da ništa nije teže i izazovnije od pisanja sistemskog softvera. Nikad nisam naišao na toliko problema kao tokom pisanja ovog editora, ali se nikada nisam ni bolje zabavljao.

Vladimir Kostić

ma (soft keys)! Nemojte. dakle, više da zavidite vlasnicima „komodora" na dirkama sa desne strane njihovog kompjutera. jer će i vaš „spektrum", kada otkucate DEFK W. „Računar" ispisivati reč 'Racunar' kada god pritisnete W. Nema prepreke ni da definišete taster Q tako da izvršava niz editorskih komandi, na primer da „otkuca" ENTER, pređe na sledeći red i pozicionira kursor na njegov kraj.

Od najboljih najbolje

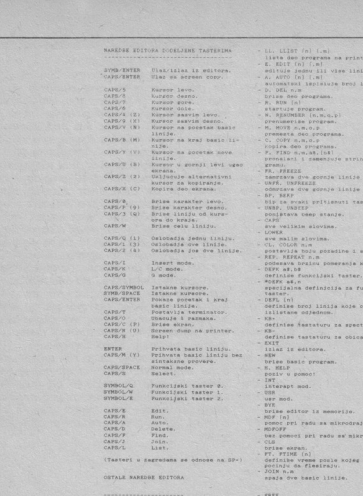
Pored editora „a la" „komodor 64", 'Ekranski editor' Vlade Kostića nudi i malo BBC-jevog imidža. Ukoliko vam je nekada potrebno da kopirate delove nekih linija na druga mesta u programu, pritiskom na samo jedan taster ćete razdvojiti kurzor u dva, koji će se nezavisno pomerati po ekranu. Upravljajući jednim od njih možete da se pozicionirate ispod bilo kog dela teksta i, pritiskajući dirku kojoj je dodeljena funkcija COPY, kopirate tekst na mesto drugog kurzora. Naravno, razdvajanjem kurzora nije isključen „osnovni" ekranski editor, što znači da možete direktno prenositi tekst iz jedne linije u drugu bez potrebe da ovu drugu prekucavate u celini. Kada smo već počeli da govorimo o sakupljanju dobrih osobina editora raznih računara, pomenimo da je ponešto preuzeto i od „galaksije": primenom komande 'freeze' možete da „zamrznete" prvih par linija ekrana i u njih unesete neki tekst koji skrolovanje neće brisati. Taj tekst može da bude neka poruka koja stalno treba da vam bude pred očima (npr. 'CHANS' je na 23531 ili Pozovi LIli u 6 sati') ili naredba koju treba kopirati na mnogo mesta u programu. Ukoliko komandi (mnoge nismo ni pomenuli) ima previše za vaše siroto pamćenje, poslužićete se dirkom H, koja će na ekranu ispisati HELP meni sa kratkim opisima delovanja raznih editorskih komandi. Da stvar bude posebno lepa, time što ćete za trenutak pogledati HELP meni nećete izgubiti sadržaj ekrana koji ispravljate; računar će ga automatski prepisati u neki slobodan segment memorije (ako takav postoji, editor će nepogrešivo detektovati ovu situaciju) i docnije ga vratiti nepromenjenog.

Pominjanjem pune memorije došli smo do osnovnog konflikta koji moraju da razreše autori raznih 'Toolkit' programa: kako sastaviti što ozbiljniji alat za razvoj drugih programa koji će sam zauzimati što manji deo dragocene memorije? 'Ekranski editor' o kome govorimo zauzima, u konačnoj verziji. 16 K operativne memorije. što znači da se sa kasete učitava za dva minuta (ako imate mikrodrajv. operacija će biti obavljena za desetak sekundi). Slažemo se sa onim što ćete reći: 16 K je dosta prostora. a dva minuta dosta vremena. Odgovor na pitanje „zašto se 'Ekranski editor' proteže na ovu drugu prekucavate u celini. Kada smo već počeli da govorimo o sakupljanju dobrih osobina editora raznih računara, pomenimo da je ponešto preuzeto i od „galaksije": primenom komande 'freeze' možete da „zamrznete" prvih par linija ekrana i u njih unesete neki tekst koji skrolovanje neće brisati. Taj tekst može da bude neka poruka koja stalno treba da vam bude pred očima (npr. 'CHANS' je na 23531 ili Pozovi LIli u 6 sati') ili naredba koju treba kopirati na mnogo mesta u programu. Ukoliko komandi (mnoge nismo ni pomenuli) ima previše za vaše siroto pamćenje, poslužićete se dirkom H, koja će na ekranu ispisati HELP meni sa kratkim opisima delovanja raznih editorskih komandi. Da stvar bude posebno lepa, time što ćete za trenutak pogledati HELP meni nećete izgubiti sadržaj ekrana koji ispravljate; računar će ga automatski prepisati u neki slobodan segment memorije (ako takav postoji, editor će nepogrešivo detektovati ovu situaciju) i docnije ga vratiti nepromenjenog.

Pominjanjem pune memorije došli smo do osnovnog konflikta koji moraju da razreše autori raznih 'Toolkit' programa: kako sastaviti što ozbiljniji alat za razvoj drugih programa koji će sam zauzimati što manji deo dragocene memorije? 'Ekranski editor' o kome govorimo zauzima, u konačnoj verziji. 16 K operativne memorije. što znači da se sa kasete učitava za dva minuta (ako imate mikrodrajv. operacija će biti obavljena za desetak sekundi). Slažemo se sa onim što ćete reći: 16 K je dosta prostora. a dva minuta dosta vremena. Odgovor na pitanje „zašto se 'Ekranski editor' proteže na ovu drugu prostoru" je verovatno vrlo blizak odgovoru na pitanje „zašto je bilo potrebno devet meseci da se ovaj program napiše". Dobre karakteristike koje program ima su zahtevale velike intervencije u osnovnom „spektrumovom" operativnom sistemu. što znači da je Vlada mogao vrlo malo da se osloni na ROM svog računara. Ponovo je napisana rutina za opsluživanje ekrana (jer originalna „spektrumova" ne podržava povećanje broja karaktera u redu). rutina za skaniranje tastature (jer „spektrum" ne prepoznaje neke sekvence tastera koje 'Ekranski editor' koristi, na primer Shift i ENTER), program za tokenizaciju i detokenizaciju naredbi (zbog kompatibilnosti sa drugim periferijskim uređajima i programima; o njoj ćemo govoriti za trenutak) i mnoge druge stvari. Sve je to, dopunjeno potrebom da editor bude pouzdan i brz (ne bi bilo prijatno čekati par sekundi za svako skrolovanje ekrana), zahtevalo dosta memorije i dosta rada.

Već smo rekli da je 'Ekranski editor' potpuno kompatibilan sa raznim hardverskim i softverskim proširenjima „spektruma". Pre svega, program će nepogrešivo raditi kako na „spektrumu" tako i na „spectrumu plus", pri čemu na ovom drugom, ako na početku rada otkucate jednu naredbu. možete slobodno da koristite i dodatne tastere. za specijalne funkcije (bilo bi bolje da program samostalno detektuje situaciju u kojoj je učitan u memoriju modela plus, ali je nešto takvo izgleda nemoguće: ne može se ni sa IN ni sa PEEK prepoznati verzija sinklerovog računara, jer su ROM-ovi potpuno jednaki). Na kaseti dobijate dve verzije programa, od kojih je jedna asemblirana tako da se učitava na početak, a druga tako da se učitava na kraj memorije, što znači da ćete ekranski editor moći da koristite zajedno sa drugim 'utility' programima na koje ste navikli. Ukoliko ste, na primer, ljubitelj 'beta bejzika', pribeći ćete jednoj od dve solucije: istovremeno koristiti oba programa na način slobodne memorije, ili koristiti ekranski editor a zatim ga „ubiti" (sa BYE a onda aktivirati beta bejzik (ovo će rešenje više primenjivati vlasnici mikrodrajva ili diska). Pri testiranju programa, pritiskajući SHIFT i ENTER, lako prelazite iz standardnog „spektrumovog" ekrana koji omogućava testiranje programa u editorski koji omogućava njegovo ispravljanje.

'Ekranski editor' vam omogućava i da pišete programe koji će se izvršavati u prisustvu nekog proširenja bejzika koje se trenutno ne nalazi u memoriji. Ukoliko, na



KAKO NABAVITI PROGRAM

U dnu ovog teksta primećujete naručžbenicu kojom možete da obezbedite svoj primerak programa ili knjiga iz naše nove Biblioteke i to u pretplati. Kupovina u pretplati je kao što to obično biva, povoljnija kako za vas tako i za nas: vama odgovaraju niže cene, a nama prilika da procenimo buduće tiraže.

Da biste nabavili naslove iz naše Biblioteke, jednostavno popunite naručžbenicu ili je, ako ne želite da oštetite svoje „Računare", prepišite na dopisnicu i pošaljite na adresu „Galaksija" (za Biblioteku programa), Bulevar vojvode Mišića 17, Beograd. Verujemo da će naručeni programi biti isporučeni do kraja godine.

Obzirom da će programi 1—5 moći da se nabave isključivo posredstvom našeg časopisa, svi poručioci će biti upisani u kompjutersku bazu podataka, što znači da će biti lično obavešteni o unapređenjima i novim verzijama programa (dokumentacije) i da će imati priliku da te verzije dobijaju ili besplatno ili uz minimalnu uplatu.

1. Ekranski editor je namenjen svima koji pišu bejzik programe. Omogućava efikasno ispravljanje softvera uz rad sa 51 slovom u redu, dva kurzora, funkcijske tastere, prenumeraciju, zamrzavanje dela ekrana, različita skrolovanja i mnogo drugih stvari. Oval 100% mašinski program je potpuno kompatibilan sa raznim proširenjima za „spektrum". Pridodata je vrlo opsežna dokumentacija. Autor Vladimir Kostić.

*2. Hiperbejzik je **unapređena verzija** pobednika prošlogodišnjeg „Galaksijinog" konkursa. Učitavši ga, bejziku vašeg „spektruma" dodajte tridesetak novih naredbi i, da stvar bude posebno lepa, i sami dobijate mogućnost da taj bejzik dalje proširujete bez ikakve potrebe da se bavite asemblerom. Pridodata je opsežna dokumentacija rađena prema standardima našeg konkursa. Autor Fischer.*

3. Velika akcija je prva kompleksna igra-avantura pisana potpuno na našem jeziku. Kao divergent treba da izvedete akciju u okupiranom gradu i prebegnete na slobodnu teritoriju. Program je vrlo dopadljivo realizovan: tehnike kompresije ekrana su omogućile da u memoriju stane veliki broj slika u boji, koje bivaju prikazane kad god dođete na neko mesto. Pridodato je kratko uputstvo. Autor Aca Radovanović.

4. Eatinglish 1 program za učenje engleskog jezika kroz igru i zabavu; rađen je po principu koji su autori nazvali „All you wish" čime je program postao „nešto kao korpa sa igračkama iz koje se vadi šta se želi i kada se želi". Eatinglish ima preko 90 slika, rečnik od 360 reči kao i lekcije koje nastavnici engleskog mogu da modifikuju. Pridodato je uputstvo rađeno prema standardima našeg Konkursa. Autori S. Milekić i D. Tanaskovski.

NARUDŽBENICA

Ovim neopozivo naručujem pouzećem sledeće programe iz „Galaksijine" biblioteke

1. Ekranski editor	**(spektrum)**	**800 din**
2. Hiper bejzik	**(spektrum)**	**700 din**
3. Velika akcija	**(spektrum)**	**400 din**
4. Eatinglish	**(spektrum)**	**500 din**
5. Trodimenzionalna graf.	**(spektrum)**	**600 din**

Takođe naručujem sledeće knjige:

5. Sve „spektrumove" rutine	**300 din**
6. Sve „komodorove" rutine	**300 din**

Ime i prezime _____

Adresa _____

Mesto _____

Potpis _____

Odgovarajući iznos ću uplatiti pošteru prilikom preuzimanja programa.

26/ekranski editor i druge bajke

primer, planirate da vaš program koristi programator EPROM-a (naš projekat iz „Računara 2" i „Računara 7"), isključićete sintaksnu proveru i jednostavno kucati *LOAD. *SAVE i slično.

Tek krajem godine

'Ekranski editor' se trenutno nalazi u fazi dovršavanja i traženja „bagova" koje jedan korisnički program sebi ne sme da dopusti. I pored toga, sa njim se već može raditi na zadovoljavajući način. Posmatrajući demonstraciju, autor ovoga teksta je video nešto što nije imao priliku da vidi ni na jednom od mnogobrojnih računara, ni amaterskih ni profesionalnih, sa kojima je radio i što ga je neobično iznenadilo; ako hoćete i sami da vidite nešto slično, otkucajte DEFL 20, zatim LIST i, kada se displej zaustavi, slovo P. Verujte, bićete iznenađeni.

Samo se po sebi razume da je prvo pitanje koje se, posle demonstracije, postavlja autoru jednog fascinantnog programa „Šta misliš da radiš sa njime?". Program će, verujemo, biti objavljen u Engleskoj i, sasvim sigurno, u Jugoslaviji. Potreba da se program potpuno debaguje, da se završe svi pravni poslovi i da se organizuje štampanje uputstva (planira se vrlo kompletna dokumentacija pisana, naravno, na našem jeziku), — srpskohrvatskom i slovenačkom. Kopiranje traka i drugi tehnički poslovi onemogućuju da prve kopije krenu pre kraja ove godine. Primerak ćete, jasno, najlakše obezbediti ako ga već sada naručite, jer će se isporuka odvijati prema principima FIFO steka.

U narudžbenici su pomenute i dve knjige: **Sve „spektrumove" rutine** Jovana Skuljana i **Sve „komodorove" rutine** Zorana Životića. Primećujete, naravno, da se radi o umecima „Računara 7" i „Računara 8" — delovima časopisa koje upravo čitate. Već vas čujemo kako kažete: Hoće dva puta da mi utrape istu stvar. Pa ipak, ova smo dva naslova dodali isključivo na zahtev čitalaca: mnogi su se žalili što se interesantno štivo kao što su naši umeci štampa na (da ne kažemo izraz) papiru. Zato ćemo, ako za to bude interesovanja, preštampati umetke na kvalitetan papir, lepo ih ukoričiti i dopuniti sadržajem i indeksom. Ovaj potez predstavlja, zapravo, probu za malu biblioteku „računara" u kojoj će se, vrlo brzo, naći puno interesantnih naslova.

Da odgovorimo, na kraju, na pitanje koje nam svi postavljaju kada čuju da se upuštamo u izdavanje programa na kasetama: „kako ćete da pobedite pirate?". Jednostavno, nećemo ni da ratujemo protiv njih. Želeći da se iz naših programa nešto i nauči, nećemo ih štititi ni na kakav način; moći ćete da ih kopirate koliko god želite. Piratovanje ćemo pokušati da umanjimo isključivo ekonomskim merama: programi neće biti skupi i biće propraćeni dobrom i opsežnom dokumentacijom. Ukoliko se odlučite da kupite naše programe od nekog pirata, verovatno ćete ostati kako bez uputstva tako i bez kvalitetnog snimka (nije lepo raditi sa editorom i ne biti siguran da neka njegova opcija, usled pogrešnog presnimavanja, neće da uništi program u memoriji!). U borbi protiv softverskog piratstva na našoj je strani još jedan argument: ako autori **dobrih** programa, pošto se njihova dela ilegalno umnožavaju, mogu za svoj rad da dobiju samo neku crkavicu, novih programa neće biti!

Dejan Ristanović

Rutine operativnog sistema

58604 tablica potrebnih sadržaja tajmera CIA čipa za postizanje standardnih brzina prenosa RS232 interfejsa

58624 IOBASE

58629 SCREEN

58634 PLOT

58648 deo CINT rutine, inicijalizacija video memorije

58784 postavljanje DFLTO i DFLTN na 3 i 0, pa zatim

58792 INITV

58804 deo GETIN za tastaturu; uzima jedan znak iz reda

58829 deo CHRIN za tastaturu; „pali" se kursor, kupi karakter iz reda i ispisuje ili posebno obrađuje ako je kontrolni karakter u pitanju. ASCII kod 131 (SHIFT+STOP) proizvodi prenošenje LOAD CHR$(13) RUN CHR$(13) sekvence u red za tastaturu i ponovni skok na 58829

58930 ulaz CHRIN za tastaturu; na osnovu stanja CRSW (208) varijable, skok na 58829, ili

58938 uzimanje ekranskog koda iz video memorije i pretvaranje u ASCII

59158 SCPNT; obrada kontrolnih karaktera (DEL, INS itd.); praktično editor C64

59953 IRQ rutina; poziva UDTIM, obrađuje treptanje kursora, omogućava pokretanje motora kasetofona (ako je pritisnut PLAY ili neki drugi) i poziva SCNKEY

60039 SCNKEY; u okviru nje:

60128 uzima ASCII kod iz tablice, pa provera stanja „auto-ponavljanje" tastera i postavljanja koda u red za tastaturu; prethodno poziva preko vektora KEYLOG (655):

60232 izbor koja od četiri tablice treba da posluži za određivanje ASCII koda, prema stanju SHFLAG (653) (bit0=1, pritisnut SHIFT, bit1=1 COMM, bit2=1 CTRL)

60289 tablica ASCII kodova, samo taster

60354 tablica ASCII kodova, taster+SHIFT

60419 tablica ASCII kodova, taster+COMm ¡**0536** tablica ASCII kodova, taster+CTRL

60484 obrada kontrolnih karaktera 14, 142, 8 i 9 (promena seta slova)

60419 tablica ASCII kodova, taster+COMM

60536 tablica ASCII kodova, taster+CTRL

60484 obrada kontrolnih karaktera 14, 142, 8 i 9 (promena seta slova)

60681 TALK

60684 LISTEN

60857 SECOND

60871 TKSA

60893 CIOUT

60911 UNTLK

60926 UNLSN

60947 ACPTR

61115 deo NMI rutina za obradu RS 232

61629 kontrolne poruke operativnog sistema: I/O ERROR, SEARCHING, SAVING itd.

61743 ispis kontrolnih poruka. Ulazni parametar ofset u Y registru za poruku iz gornje tablice.

61758 GETIN

61783 CHRIN

61898 CHROUT

61968 CHKIN

62032 CHKOUT

62097 CLOSE

62223 pretraživanje tablice logičkih adresa i resert STATUS varijable.

62239 prenos iz tablica LA,SA,FA u odgovarajuće sistemske promenjive

62255 CLALL

62259 CLRCHN

62282 OPEN

62421 SDNAM

62473 otvaranje RS232 bafera i postavljanje brzine prenosa

62622 LOAD

62648 LOAD sa diska

62777 LOAD sa kasetofona

62941 SAVE

62970 SAVE na disk

63042 DCLOSE

63071 SAVE na kasetofon

63131 UDTIM

63197 RDTIM

63204 SETTIM

63213 STOP

63227 tablica maskiranih LDA instrukcija; ulaz na svaku sledeću za tri veću adresu daje LDA za jedan broj veći; rutinu koriste sve za komunikaciju za prijavljivanje greške određenog broja; odmah se izvodi i CLRCHN i greška ispisuje ako je to omogućeno stanjem MSGFLAG

63276 upis bloka zaglavlja sa trake, provera tipa datoteke i ispis naziva

63338 kreiranje bloka zaglavlja za traku

63471 poređenje naziva datoteke sa trake i zadatog

63511 kontrola stanja tastera na kasetofonu, ispis PRESS PLAY . . . i OK

63544 rutine za vremensku sinhronizaciju slanja podataka na kasetofon, kontrolu motora itd.

63788 rutine za čitanje sa trake

64422 rutine za prenos podataka na traku

64738 START

64770 AOINT

64789 RESTORE

64794 VECTOR

64816 tablica vektora operativnih sistema

64848 RAMTAS

64931 IOINIT

65017 SETNAM

65024 SETLFS

65031 READST

65048 SETMSG

65061 MEMTOP

65076 MEMBOT

65091 NMI rutina i provera STOP+RESTORE

65126 BRK reset

65352 IRQ ulaz; ispituje da li je interapt nastao spolja ili je izvedena BRK instrukcije.

65371 CINT

65408 broj verzije operativnog sistema

65409 KERNAL JUMP TABLICA

65530 NMI vektor

65532 RESET vektor

65535 IRQ vektor

Zoran Životić

Sve „komodorove" rutine

organizacija računara i potprogrami iz ROM-a

Sistemski program računara „komodor 64" zauzima 16 K ROM memorije i podeljen je u dva dela: OPERATIVNI SISTEM i BEJZIK INTERPRETER. Fizički su odvojeni u memorijske blokove od 8K, ali bejzik koristi deo drugog bloka, tako da sam operativni sistem zauzima nešto manje od 7 K. Za razliku od mnogih drugih kućnih računara, može se postaviti jasna granica između ove dve funkcionalne celine.

Mnogi ljubitelji računara su navikli na prikaze u kojima se do poslednjeg bajta „rastura" ROM, sa svim potprogramima čijim se pozivom može uštedeti makar i jedna instrukcija u vlastitom programu. Podstičući hakere na ovakav pristup, konstruktori računara, koji su takve programe na sličan način i sastavljali, olakšali su potpuno razumevanje rada računara i programa koji ga kontroliše. To je, međutim, primer na kome ne treba učiti.

Jasno odvojivši celine i dosledno se pridržavajući usvojene koncepcije, „Komodor" je proizveo računar čiji operativni sistem sigurno nije vrhunski domet, ali pruža mogućnost jednostavnog pisanja programa. Zbog toga smo prinuđeni da u ovom prikazu poštujemo takav stav i damo pregled po zvaničnim rutinama kako ih je definisao „Komodor". Autor ovog priloga mora da prizna da spada u one koji su već rekli dan po kupovini računara izlistali svih 16 K mašinskog programa i da je prvo pokušao za ovaj pregled da operativni sistem razbije na ranije pomenuti način, ali je odustao. Mada takav prikaz ne bi bio ništa složenији, potpuno bi uništio jasnoću pregleda i ono na čemu „Komodor" posebno insistira: kompatibilnost programa sa budućim verzijama njihovih računara. Sto već u ovom trenutku postaje aktuelno pojavom C 128. Nastojali smo da povremeno damo objašnjenja šta se u samoj rutini dešava da bismo ohrabrili one koji misle da je „sve to jako komplikovano". Međutim, da bi mogao iole ozbiljnije da petlja sa ROM-om, programer treba prvo da bude upućen u organizaciju računara i njegove memorije i, naravno, sistemske promenljive.

Na samom kraju dat je kratak pregled, bez posebnih objašnjenja, rasporeda zvaničnih i nezvaničnih delova operativnog sistema sa direktnim pozivnim adresama, zbog onih koji sami žele da pregledaju kako su napisani i tako steknu iskustva za pisanje sličnih programa.

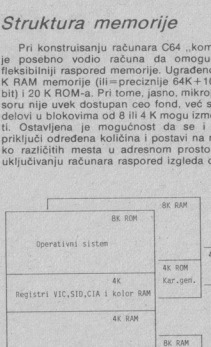
„Komodor 64" se sastoji od nekoliko osnovnih komponenti:

- mikroprocesor 6510
- 64 K RAM memorije (8×4164)
- 20 K ROM memorije
- video procesor VIC 6567
- generator zvuka SID 6581
- kompleksni interfejs adapteri 2×CIA 6526

Poslednje tri komponente sa ovog spiska su posebno interesantne. Radi se o složenim čipovima sa kojima bejzik praktično ne ostvaruje nikakav kontakt. Jedan, manji deo, CIA je iskorišćen za komunikaciju sa periferijama, dok je ostatak slobodan za „bockanje" neizbežnim POKE i PEEK. Ujedno su i informacije o korišćenju ovih čipova često komplikovane, pa i pogrešne. Stoga nije čudno da većina vlasnika ne zna, na primer, da u njegovom računaru postoje dva kompletna, kako se to popularno kaže, digitalna časovnika sa alarmom, koji prosto čeznu da budu pokrenuti. Oni koji žele da koriste računar za upravljanje „spoljnim svetom" imaju gotovo sve što im je potrebno, pa je za jednostavnije primene dovoljno nabaviti samo odgovarajući konektor. Dva „komodora" mogu bez ikakvih dodataka da komuniciraju među sobom, a potrebne su minimalne investicije da ostvare vezu sa bilo kojim većim računarom.

Struktura memorije

Pri konstruisanju računara C64 „komodor" je posebno vodio računa da omogući što fleksibilniji raspored memorije. Ugrađeno je 64 K RAM memorije (ili=preciznije 64K+1024×4 bit) i 20 K ROM-a. Pri tome, jasno, mikroprocesoru nije uvek dostupan ceo fond, već se neki delovi u blokovima od 8 ili 4 K mogu izmenjivati. Ostavljena je mogućnost da se i spolja priključi određena količina i postavi na nekoliko različitih mesta u adresnom prostoru. Po uključivanju računara raspored izgleda ovako:



STOP 65505 $FFE1

TESTIRANJE STOP TASTERA

A:#:-
X:#:-
Y: :-
S:#:Z=1 ako je pritisnut

Naziv ove rutine može da zavara. Ona ne ispituje stvarno stanje, već varijablu STKEY kojoj se vrednost dodeljuje u UDTIM, pa u slučajevima kada je interapt isključen treba pozvati obe. Ako je STKEY=127, znači da je za vreme UDTIM taster STOP bio pritisnut, pa se pre povratka izvodi i CLRCHN.

SCNKEY 65439 $FF9F

OČITAVANJE TASTATURE

A:#:-
X:#:-
Y:#:-
S:#:-

PR :ako je preskočena normalna
inicijalizacija obavezno IOINIT

Rutina se poziva u normalnoj interapt proceduri. Ako je neki od tastera pritisnut, njegov ASCII kod se smešta u bafer za tastaturu od adrese 631, koji je organizovan kao red. Prvi ASCII kod koji je u njega stavljen biće pročitan pozivom GETIN rutine, a svi ostali zatim pomereni jedno mesto ka početku. Broj tastera u redu koji čeka na obradu određuje varijabla NDX (198), dok ih najviše može biti prema vrednosti variјable XMAX (649), ali ne više od 10. Kada se red napuni, svi dalji tasteri se ignorišu.

Skaniranje tastature daje kao rezultat vrednosti između 0 i 64 koje rutina prevodi u ASCII kod, vodeći pri tome računa o stanju SHIFT, C i CTRL tastera. ASCII kodovi su smešteni u 4 tablice u kojima se pomenuta vrednost od 0—64 koristi kao ofset. Adrese tablica su:

60288 – samo taster
60354 – taster + SHIFT
60419 – taster + COMM
60536 – taster + CTRL

Kada pritisnete, na primer DEL, rutina dobija rezultat skaniranja 1. Na prvom mestu u prvoj tablici nalazi se broj 20, što je odgovarajući ASCII kod. Posebno je interesantno pogledati četvrtu tablicu koja sadrži kodove za slučaj istovremenog pritiska CTRL tastera. Na dosta mesta se nalazi broj 255, što znači da takva kombinacija ne daje nikakav rezultat (na primer CTRL+DEL), ali zato sva slova i znak @ imaju odgovarajuće kodove od 0 do 26 abedecnim redom. Znajući to, nema potrebe za korišćenjem CHR$ funkcije u bejziku za ispis kontrolnih znakova. Ako štampač zahteva za prelazak u grafički mod znak 8, između znakova navoda stringa koji šaljete, otkucajte CTRL+H.

Postoji još jedna pogodnost u ovoj rutini. Deo, koji na osnovu stanja SHIFT, C i CTRL tastera, odlučuje koju tablicu od četiri navedene upotrebiti za nalaženje ASCII koda, poziva se indirektno preko vektora KEYLOG (655,656). Izmenom vektora moguće dopisati slično i kreirati naše tablice kodova, te tako promeniti raspored slova na tastaturi. Jedna od primena može biti zamena nekog nepotrebnog grafičkog znaka našim slovom Č, a zatim definisanja tastera * sa kodom zamenjenog znaka. Dobili smo naše slovo na standardnom mestu, a nismo izgubili znak množenja, već ga možemo premestiti na neku drugu poziciju, na primer SHIFT+=.

PLOT 65520 $FFF0

ČITANJE/POSTAVLJANJE POZICIJE KURSORA

A:#:-
X:#:red
Y:#:kolona
S:#:C=1 čitanje,C=0 postavljanje

Rutina nudi proizvoljno postavljanje (ili očitavanje) trenutne pozicije kursora, sa čim se, inače, bejzik dosta muči. Verovatno još jedan dokaz da je „Komodor" više razmišljao o onima koji se bave pravljenjem komercijalnih programa nego o povremenim korisnicima bejzika.

```
        SEC
        JSR  PLOT
        CPX #10
        BCC  OK
        LDX #0
        LDY #255
OK      INY
        CLC
        JSR PLOT
```

UNTLK

65451 $FFAB

NAREDBA SI UREDJAJIMA DA PRESTANU SA SLANJEM

```
A:#:-
X: :-
Y: :-
S:#:-

PR :-
GR :STATUS
```

UNTLK ima dejstvo na sve uređaje istovremeno i nakon nje odmah prestaju sa slanjem podataka računaru.

```
        LDA #8
        JSR   TALK
        JSR   READST
        BNE   ERROR
        LDA #%10110111   ;SA=15
        JSR   TKSA
        LDY #0
INPUT   JSR   ACPTR
        STA   512,Y
        INY
        CMP #13
        BNE   INPUT
        JSR   UNTLK
```

Obrada prekida

Komunikacija sa periferijama, u koju smo uračunali i editor (ceo editor je sadržan u rutini CHRIN), i inicijalizacija iscrpljuju zadatke operativnog sistema. Ostaje svega nekoliko rutina koje se koriste uglavnom u interapt sekvenci.

SETTIM

65499 $FFDB

POSTAVLJANJE ČASOVNIKA REALNOG VREMENA

```
A: :viši bajt
X: :sledeći niži
Y: :sledeći niži
S:#:-
```

Tri bajta u nultoj strani su odvojena za brojač čija se vrednost povećava pri svakom interaptu za očitavanje tastature. Na osnovu njega bejzik generiše varijable TI i TI$. Ovom se rutinom brojač može postaviti na proizvoljnu vrednost. U nazivu rutine reč „realnog" treba prihvatiti sa mnogo rezerve: brojač će stati ako se spreči interapt.

Interesantno je da C64 ima u svoja 2 CIA čipa prave časovnike sa alarmom koji može generisati interapt, ali se uopšte ne koriste.

```
        LDA #0
        TAX
        TAY
        JSR   SETTIM
WAIT    JSR   RDTIM
        CPX #5
        BCC   WAIT
```

RDTIM i SETTIM se mogu upotrebiti za pravljenje dužih pauza u programu, kada neku poruku treba zadržati na ekranu i slično.

RDTIM

65502 $FFDE

OČITAVANJE ČASOVNIKA REALNOG VREMENA

```
A:#:viši bajt
X:#:sledeći niži
Y:#:sledeći niži
S:#:-
```

UDTIM

65514 $FFEA

AŽURIRANJE ČASOVNIKA REALNOG VREMENA

```
A:#:-
X:#:-
Y: :-
S:#:-
```

Ova rutina je deo programa za obradu interapta i povećava vrednost brojača za 1. Ako je dostignuta granica od 72271 (oko 20 min.), brojač se postavlja na nulu. Međutim, rutina ima jedan još značajniji zadatak: izdvojeno očitavanje stanja STOP tastera. Ako je u tom trenutku pritisnut, vrednost sistemske promenjive STKEY (145) će biti postavljena na 127.

Svaki ROM ima svog RAM dvojnika, ali je stvar nešto komplikovanija u delu ispod operativnog sistema. U adresnom prostoru su registri čipova, dok se „ispod" nalaze generator znakova i 4K RAM memorije. Pristup ROM-u u normalnom radu je potreban samo video procesoru, koji mu se obraća bez posredovanja mikroprocesora, pa zbog toga i nije neophodno da bude u memorijskoj mapi. Ipak je ostavljena i mogućnost da mu se sadržaj čita. Time je skraćen postupak pri oblikovanju različitih setova slova.

Za obraćanje memoriji (POKE,PEEK ili nekom od mašinskih instrukcija) ugrađena je i jedna posebna pogodnost. Pošto i operativni sistem i bejzik imaju u zaleđini odgovarajuću količinu RAM-a, svaki upis u neku od ovih lokacija (što inače ne bi imalo efekta jer se radi o ROM-u) će izvršiti upis u RAM ispod, dok se čitanje memorije odnosi na onu koja je trenutno u adresnom prostoru. Tako operativni sistem može da upiše neki drugi „ispod" sebe i da mu onda prepusti kontrolu. Jednu od mogućih primena ilustruje i sledeći primer:

```
10 FOR F=40960 TO 49151: POKE
F,PEEK(F): NEXT F
20 POKE 41118,75:POKE 41119,82:POKE
41120,65:POKE 41121,202
30 POKE 41122,90:POKE 41123,193
40 POKE 1,54:END:END:END
RUN
LIST
```

U adresnom prostoru se može naći svaki od memorijskih blokova ali ne i u proizvoljnim kombinacijama. Kontrola se ostvaruje preko 3 linije izlaznog porta mikroprocesora. On je ugrađen u mikroprocesor (to je i jedina razlika u odnosu na 6502 koji ga nema) i prekidapa memorijske adrese 1 i 0. Na nultoj lokaciji se nalazi registar za definisanje smera prenosa podataka za svaki bit porta koji je na adresi 1. Ako je neki bit jednak 1, onda će odgovarajuća linija na adresi 1 biti izlazna. Prve tri linije se koriste za raspoređivanje memorije, pa su i odgovarajući bitovi u registru za smer postavljeni na 1. Ostale tri linije kontrolišu kasetofon:

ADRESA	BIT 5	BIT 4	BIT 3
0!	1	0	1
1!	Uključ. isključ. motora	Registrovanje pritiska kas. tastera	Linija za prenos podataka

ADRESA	BIT 2	BIT 1	BIT 0	RASPORED MEMORIJE
0!	1	1	1	
1!	1	1	1	Standardni raspored
1!	1	1	0	BASIC ROM zamenjen sa 8K RAM
1!	1	0	1	BASIC ROM i KERNEL ROM zamenjeni sa 16K RAM
1!	0	1	1	Karakter generator ROM u adresnom prostoru
1!	1	0	0	Svuda 64K RAM

Postoje i druge kombinacije, ali se one odnose na uvođenje dodatnih signala spolja, kada se priključuje dodatni ROM (ili EPROM, RAM) sa gotovim programom, čime se ovde nećemo baviti.

Ako umesto operativnog sistema ili bejzika uvodite RAM, podrazumeva se da u njemu mora postojati program koji će preuzeti kontrolu nad računarem. Kada se iz adresnog prostora izbacuju registri CIA, mora se biti posebno pažljiv. Oni su odgovorni za generisanje interapta za očitavanje tastature i ostale sistemske operacije neophodne za normalan rad. Da bi se pročitao ROM sa generatorom znakova, potrebno je sve ove funkcije privremeno isključiti, što će biti objašnjeno u prikazu samog CIA čipa.

VIC — video procesor

Video procesor ima mnogo različitih mogućnosti, koje (čini nam se i previše) pa je bilo neophodno da se stvari malo iskomplikuju. To se posebno odnosi na mogućnost izbora različitih delova memorije za definisanje slike visoke rezolucije, sprajtova itd. Ipak, stvari postaju nešto jasnije ako se ima na umu da VIC posmatra memoriju potpuno nezavisno od mikroprocesora, tako da sve što će odnositi na memorijski raspored koji „vidi" mikroprocesor ne važi i za njega. Ako se izabere da slika visoke rezolucije bude u poslednjih 8K, nije neophodno da se ROM zamenjuje RAM-om; sem je dizajniran tako da VIC na tom mestu „vidi" RAM bez obzira šta je u adresnom prostoru mikroprocesora. Međutim, mikroprocesor u taj RAM mora da postavi vrednosti koje će dati traženu sliku, pa je to pitanje koje se odvojeno posmatra. U ovom slučaju, dovoljno je upotrebiti POKE, jer će vrednost biti zapisana u RAM, kako je to ranije objašnjeno, iako je ROM operativnog sistema aktivan. Ako bi želeli da ispitujemo sadržaj pojedinih delova slike upotrebom PEEK, morala bi da se izvrši zamena. Za sve to vreme VIC na ekranu daje normalnu sliku visoke rezolucije.

Adresa za definicije sprajtova

Sprajt se definiše u bloku od 64 bajta. Zadavanje početne adrese je vezano za video memoriju. Poslednjih 8 bajtova koji se ne koriste (40×25=1000, što ostavlja 24 bajta slobodno) sadrže apsolutnu adresu od koje počinje definicija odgovarajuće sprajta, podeljenu sa 64 (mora biti deljiva). Tako je za sprajt 0, ako se kreira od adrese 12800, potrebno na adresu 2040 staviti broj 200 (12800/64=200). Kada se video memorija premešta, i dalje sa sobom nosi registre za adresu sprajta u svojih poslednjih 8 bajtova, pa će adresa 2040 iz gornjeg primera biti drugačija.

Za sprajtove važi isto što i za visoku rezoluciju: definicije se ne mogu preklapati sa generatorom znakova, pa se adrese sprajta od 64 do 127 ne mogu koristiti.

Registar za prekide

Korisna osobina video procesora je i mogućnost generisanja interapta. Bit registra

53273 će biti postavljen na 1 ako se neki od događaja za koji je zadužen desio (vidi pregled registara). U tom stanju će ostati sve dok se na njegovo mesto ne upiše bit 1, čime će biti obrisan i spreman za ponovno registrovanje događaja (još jednom da naglasimo: iako su u memorijskoj mapi, kontrolni registri se često ponašaju sasvim drugačije od memorije). Međutim ovo nije dovoljno da bi se generisao interapt. Potrebno ga je omogućiti postavljanjem odgovarajućeg bita u registru 53274 na 1. Nakon toga, mikroprocesor će biti prekinut i izvršen skok na interapt rutinu preko vektora CINV. Na ovaj način se mogu jednostavno realizovati igre, jer nije potrebno vršiti proveru da li su, na primer, dva sprajta u sudaru. Vektor CINV se može izmeniti da pokazuje na rutinu za obradu ovog događaja dok se samo registrovanje prepušta VIC-u. U slučaju pojave interapta, bit 7 registra će biti postavljen na 1. Pošto i CIA može generisati prekid, u programu se, proverom ovog bita, može utvrditi da li je prekid generisao VIC ili neki drugi izvor.

Registar rastera

Na ekranu monitora slika se ostvaruje stalnim kretanjem elektronskog mlaza u horizontalnim linijama. Ovo kretanje kontroliše VIC, pa je u njega ugrađen brojač ispisanih linija. Njegov 8 bit (linija ima više od 256) se nalazi u registru 53265. Raster registar ima dvostruku namenu. Pri čitanju pokazuje koja se linija trenutno ispisuje (polje ekrana na kome se nalaze znakovi počinje od linije 51, a prestaje na 251), dok se pri upisu zapisana vrednost čuva i u svakom ciklusu ispisivanja slike poredi sa trenutnim stanjem brojača. Kada se vrednosti poklope, postavlja se nulti bit interapt registra na 1, a ako je omogućen, biće generisan i interapt. Sve se ovo dešava vrlo brzo, ali i dovoljno sporo za kontrolu iz mašinskog jezika. Na primer, upišete u raster registar broj 151 i omogućite interapt postavljanjem bita 0 u registru 53274 na 1. Kada interapt nastupi, u rutini za njegovu obradu prebacite VIC u mod za visoku rezoluciju i u raster registar upišete broj 251. Pri sledećem interaptu vratite VIC u tekst mod, a raster postavite opet na 51. Rezultat: prva polovina ekrana u tekst modu, a druga u visokoj rezoluciji. Na isti način se može proizvesti utisak da je na ekranu istovremeno preko 70 sprajtova i slično. Promena se odvija veoma brzo, pa treba onemogućiti ostale izvore prekida, jer prelaz između dve slike može biti nestabilan.

Fino pomeranje slike

Promenom sadržaja prva tri bita registara 53265 i 53270 slika se na ekranu može pomeratiu svim pravcima po jednu ekransku liniju maksimalno 8 mesta, što je, ujedno, i veličina jednog znaka. Promenom vrednosti od 0 do 7 slika se blago pomera, pa onda izvrši stvarno premeštanje znaka iz jednog reda u drugi i vrednost ponovo menja od 0—7. Dobija se efekat klizanja slike bez skokova. Ako se izabere veličina slike 24×38 (isti registri, bit 3), krajnje linije se neće videti. U njih se smeštaju podaci koji tek treba da se pojave na ekranu da bi kasnije blago „isplivali" u vidljivo područje.

```
49152                   **  49152
49152 120               SEI
49153 169 033           LDA #<START
49155 162 192           LDX #>START
49157 141 020 003       STA 788
49160 142 021 003       STX 789
49163 169 051           LDA #51
49165 141 018 208       STA 53266
49168 133 002           STA 2
49170 169 001           LDA #1
49172 141 026 208       STA 53274
49175 173 017 208       LDA 53265
49178 041 127           AND #%01111111
49180 141 017 208       STA 53265
49183 088               CLI
49184 096               RTS
49185 120       START   SEI
49186 174 025 208       LDX 53273
49189 048 003           BMI YES
49191 076 049 234       JMP 59953
49194 165 002   YES     LDA 2
49196 201 051           CMP #51
49198 240 006           BEQ L1
49200 169 051           LDA #51
49202 160 056           LDY #56
49204 208 004           BNE L2
49206 169 151   L1      LDA #151
49208 160 024           LDY #24
49210 133 002   L2      STA 2
49212 141 018 208       STA 53266
49215 140 017 208       STY 53265
49218 142 025 208       STX 53273
49221 104               PLA
49222 168               TAY
49223 104               PLA
49224 170               TAX
49225 104               PLA
49226 064               RTI
```

Ekran visoke rezolucije zahteva 8K memorije, ekran niske rezolucije traži 1K, sprajtovi najviše 1/2K, pa je za sve informacije video procesoru dovoljno 16K. Kako „komodor" ima 64K, postoje četiri mesta koja bi VIC mogao da odvoji za svoje potrebe. Konstruktori su izabrali da mu dodele prvih 16 kilobajta, ali je ostavljena mogućnost da se ovo izmeni u toku rada i izabere bilo koji od četiri moguća bloka. Ekran niske rezolucije zauzima 1 kilobajt od adrese 1024, dok je RAM memorija koja sadrži informaciju o boji svakog znaka potpuno izdvojena i nepromenjivo smeštena od adrese 55296. U ovako izabranom rešenju pojavljuje se problem: gde smestiti 4K generatora znakova? Konstruktori su ga postavili od adrese 4096, ali problem nastaje kad želimo iz bejzika da pročitamo njegov sadržaj. On bi se tada morao prebaciti u adresni prostor mikroprocesora, ali onda se gubi RAM na tom mestu, a sa njim, najverovatnije (ako je nešto veći), i sam program koji treba da obavi čitanje! Zato je pribegnuto lukavstvu: kada VIC traži informaciju sa tog memorijskog prostora, on stvarno koristi adrese od 4096; mikroprocesor za isto

UNLSN

65454 $FFAE

NAREDBA SI UREDJAJIMA DA PRESTANU SA SLUŠANJEM

A:#:-
X: :-
Y: :-
S:#:-

PR :-
GR :STATUS

Rutina nema nikakvih ulaznih parametara jer se svim uređajima na SI istovremeno naređuje da prestanu sa slušanjem.

```
LDA #4
JSR LISTEN
LDA #%01100111  ;SA=7
JSR  SECOND
LDA #48
JSR  CIOUT
LDA #13
JSR  CIOUT
JSR  UNLSN
```

DCLOSE

63042 $F642

ZATVARANJE DATOTEKE

A:#:-
X:#:-
Y:#:-
S:#:-

PR :SETLFS
GR :STATUS

Ulazni parametri rutine su prva i sekundarna adresa koje se moraju naći u sistemskim promenljivim FA (186) i SA (185). Ako između otvaranja datoteke i njenog zatvaranja nije korišćena ni jedna druga veza, nema potrebe pozivati SETLFS, jer su FA i SA očuvani. U sekundarnoj adresi bit 7 mora biti 0, inače se rutina odmah napušta.

TALK

65460 $FFB4

NAREDBA SI UREDJAJU DA POČNE SA SLANJEM

A:#:broj uredjaja
X: :-
Y: :-
S:#:-

PR :-
GR :STATUS

TALK inicijalizuje prenos podataka ka računaru. Kada uređaj primi naredbu TALK, zahteva da mu se prvo pošalje sekundarna adresa,i tek onda postavlja podatke na SI. Slanje sekundarne adrese obavlja TKSA.

TKSA

65430 $FF96

SLANJE SEKUNDARNE ADRESE NAKON TALK

A:#:sekundarna adresa
X: :-
Y: :-
S:#:-

PR :TALK
GR :STATUS

Sekundarna adresa kao i u ostalim slučajevima mora imati bitove 5 i 6 setovne.

ACPTR

65445 $FFA5

UPIS BAJTA SA SI

A:#:primljeni bajt
X:#:-
Y: :-
S:#:-

PR :TALK,TKSA
GR :STATUS

Kada spoljnji uređaj postavi bajt na SI, ova rutina će ga pročitati i vratiti u akumulatoru. Ako podatak nije poslat, rutina će čekati samo određeno vreme i zatim javiti grešku.

Serijski interfejs

Iako sve rutine za komunikaciju sa periferijama u sebi sadrže i ove potprograme, „Komodor" je ipak omogućio njihovo izdvojeno korišćenje. Time otpada potreba poštovanja procedura operativnog sistema za otvaranje i zatvaranje veze, ali otpadaju i poruke o greškama. STATUS varijabla je jedina informacija o stanju komunikacije.

Svakvim pristupom je otvorena mogućnost dopuna operativnog sistema. Ako se tome doda vektorsko pozivanje rutina OPEN, CLOSE, CHKOUT i ostalih, mogućnost zamene bejzik ROM-a sa RAM memorijom, „komodor 64" počinje više da liči na „poligon" za razvoj sistemskih programa nego na „običan" kućni računar.

Pri upotrebi ovih rutina posebnu pažnju treba obratiti na stanje pojedinih bitova u sekundarnoj adresi, jer je ono od presudne važnosti za pravilan rad.

LISTEN

65457 $FFB1

NAREDBA SI UREDJAJU DA POČNE SA SLUŠANJEM

```
A:#:broj uredjaja
X: :-
Y: :-
S:#:-

PR :-
GR :STATUS
```

Ovom se rutinom spoljnjem uređaju komanduje smer komunikacije. Nakon LISTEN, on je spreman za prihvatanje podataka, pri čemu prvi treba da bude sekundarna adresa. Za njeno slanje služi sledeća rutina:

SECOND

65427 $FF93

SLANJE SEKUNDARNE ADRESE NAKON LISTEN

```
A:#:sekundarna adresa
X: :-
Y: :-
S:#:-

PR :LISTEN
GR :STATUS
```

Sekundarna adresa, kao i broj uređaja, šalje se sa posebnima stanjem kontrolnih linija SI, pa se tako razlikuje od običnog bajta poslatog na vezu. Vrednost može biti od 0—15, a bit 6 i 5 moraju biti 1. Kada se otvara datoteka na disku, moraju i bitovi 7 i 4 takođe biti 1, posle čega se šalje naziv CIOUT rutinom. Cela procedura otvaranja datoteke na disku obejdinjena je u rutini SDNAM koju koristi OPEN.

Da bi se datoteka na disku zatvorila i unele odgovarajuće promene u direktoriju, potreban je sličan postupak; prvo se izvodi LISTEN a zatim i SECOND, ali je sada bit 4 sekundarne adrese 0. Nakon UNLSN datoteka je zatvorena. Procedura je objedinjena u DCLOSE, koju koristi rutina CLOSE.

CIOUT

65448 $FFA8

ISPIS BAJTA NA SI

```
A: :bajt koji se šalje
X: :-
Y: :-
S:#:-

PR :LISTEN,SECOND
GR :STATUS
```

Bajt poslat ovom rutinom primiće svaki uređaj kome je prethodno zadat LISTEN.

SDNAM

62421 $F3D5

SLANJE SEKUNDARNE ADRESE I NAZIVA DATOTEKE

```
A:#:-
X:#:-
Y:#:-
S:#:*

PR :SETNAM-SETLFS
GR :5-STATUS
```

Ova rutina otvara datoteku na samom spoljnjem uređaju. Izvodi se LISTEN i SECOND, pri čemu se poslednja 4 bita sekundarne adrese postavljaju na 1. Ako je bit 7 pre toga bio 1, rutina će odmah biti napuštena. Zatim se šalje naziv datoteke ako postoji (dužina > 0> . Upotreba ove rutine u sebi krije i jednu opasnost; ako uređaj nije priključen (nakon LISTEN bit 7 STATUS-a = 1), izvođe se dve PLA instrukcije čime je povratna adresa izgubljena.

MOD	UKLJUČIVANJE		INFORMACIJE O BOJI
TEKST	inicijalno		53281=boja pozadine svih karaktera (0-15) CRAM =boja pojedinačnog karaktera (0-15)
TEKST VIŠEBOJNI KARAKTERI	53270	bit4=1	53281=boja pozadine svih karaktera (0-15) 53282=boja para bitova 01 u def.kar.(svi karakteri) 53283=boja para bitova 10 u def.kar.(svi karakteri) CRAM =boja para bitova 11 u def.kar.(pojedinač. 0-7 Ako je CRAM=(8-15) karakt.jednobojan (boja 0-7)
TEKST VIŠEBOJNA POZADINA	53265	bit6=1	53281=pozadina karaktera sa ekranskim kodom 0- 63 53282=pozadina karaktera sa ekranskim kodom 64-127 53283=pozadina karaktera sa ekranskim kodom 128-191 53284=pozadina karaktera sa ekranskim kodom 192-255 CRAM =boja samog karaktera (0-15) pojedinačno U sva 4 slučaja se vide karakteri sa ek.kod. 0-63
VISOKA REZ. 320x200 2 boje	53265	bit5=1	Svaka od 1000 lokacija VIDEO MEMORIJE kontroliše boju tačke i pozadine u kvadratu 8x8 (velicina jed. karaktera u tekst modu) bit 0-3=boja pozadine (0-15) bit 4-7=boja tačke (0-15)*16
VISOKA REZ. 160x200 4 boje	53265 53270	bit5=1 bit4=1	53281=boja cele pozadine VIDEO MEM bit 0-3=boja para bitova 10 u polju 8x8 bit 4-7=boja para bitova 01 u polju 8x8 CRAM =boja para bitova 11 u polju 8x8 Par bitova se vidi kao jedna tačka

VIC

ADRESA REGIST.	BIT 7	6	5	4	3	2	1	0
53248			SPRAJT 0	X POZICIJA				
53249			SPRAJT 0	Y POZICIJA				
53250			SPRAJT 1	X				
53251			SPRAJT 1	Y				
53252			SPRAJT 2	X				
53253			SPRAJT 2	Y				
53254			SPRAJT 3	X				
53255			SPRAJT 3	Y				
53256			SPRAJT 4	X				
53257			SPRAJT 4	Y				
53258			SPRAJT 5	X				
53259			SPRAJT 5	Y				
53260			SPRAJT 6	X				
53261			SPRAJT 6	Y				
53262			SPRAJT 7	X				
53263			SPRAJT 7	Y				

53264	SPRAJT 7 - 0				8. BIT X POZICIJE			
53265	raster regist. bit 8	tekst sa 4 boje	visoka rezolu-cija	slika nevid. 0=nevid	24/25 redova 1=25	fino skrolovanje za 0 do 7 delova linije po Y koordinati		
53266	RASTER REGISTAR BIT 7-0							
53267	SVETLOSNO PERO X KOORDINATA							
53268	SVETLOSNO PERO Y KOORDINATA							
53269	7	6 (pojava sprajta 1=da)			5	4	3	2	1	0
53270	x	x	uvek mora 0	višebojni mod 1=da	38/40 kolona 1=40	fino skrolovanje za 0 do 7 delova linije po X koordinati		
53271	7	6 (sprajt povećan 2x vertik. 1=da)					1	0
53272	Adresa video memorije				Adresa karakt. generat.			x
53273	1=bilo koji dozvoljeni	IZVOR INTERAPTA x	x	x	svetlo. pero	sudar dva sprajta	sudar sprajta i pozad.	dostig. zadati raster
53274	DOZVOLA INTERAPTA (1=DA) OD				-"-	-"-	-"-	-"-
53275	7	(1=prioritet sprajta nad pozad.)			2	1	0	
53276	7	(1=višebojni sprajt)			3	2	1	0
53277	7	6 (sprajt povećan 2x horizont. 1=da)					1	0
53278	7	(sprajt u sudaru sa neki drugim sp. 1=da)					1	0
53279	7	(sprajt u sudaru sa pozadinom 1=da)			2		1	0
53280	BOJA RAMA SLIKE							
53281	BOJA POZADINE							
53282	BOJA POZADINE 1							
53283	BOJA POZADINE 2							
53284	BOJA POZADINE 3							
53285	VIŠEBOJNI SPRAJT BOJA 1							
53286	VIŠEBOJNI SPRAJT BOJA 2							
53287	OSNOVNA BOJA SPRAJTA 0							
53288	SPRAJTA 1							
53289	2							
53290	3							
53291	4							
53292	5							
53293	6							
53294	7							

čuje zatvaranje svih veza u programskoj petlji bez potrebe da budu redom numerisane.

Kada je veza otvorena za kreiranje datoteke na kaseti, ako je sekundarna adresa bila 2 ili 3 (vidi tablicu sekundarnih adresa kod SETLFS). rutina CLOSE će zapisati još jedan blok podataka sa prvim bajtom 5, kao oznakom „kraj trake".

LOAD

65493 $FFD5

UPIS U SEGMENT MEMORIJE/PROVERA ZAPISA

A:#:0=LOAD, 1=VERIFY
X:#:niži bajt adrese
Y:#:viši bajt adrese
S:#:*

PR :SETLFS,SETNAM
GR :0-4-5-8-9-STATUS

Ulazni parametri zavise od izbora sekundarne adrese. Ako je SA=1, sadržaji X i Y registra nisu važni. Program će biti upisan na izvornu adresu. Ako se zada SA=0, onda X Y moraju da sadrže početnu adresu od koje se program smešta u memoriju. Pri povratku iz rutine, u njima će se naći poslednja adresa do koje se stiglo u upisu.

Ove varijante sa sekundarnom adresom nemaju veze sa operativnim sistemom u disk jedinici. Disk za LOAD uvek zahteva SA=0 (odnosno 01100000), što mu računar i šalje. Interna je stvar rutine da na osnovu zadate SA odluči da li da ignoriše prva dva bajta iz programa (izvorna adresa), ili da ih iskoristi kao početnu adresu.

```
        JSR  SETLFS
        LDA #%01000000
        LDA #7
        LDX #<NAME
        LDY #>NAME
        JSR  SETNAM
        LDX #1
        LDY #0
        JSR  SETMSG
        LDA #0
        LDX #1
        LDY #8
        JSR LOAD
        ...
NAME    .ASC "OLIMPIC"
```

SAVE

65496 $FFD8

SNIMANJE SEGMENTA MEMORIJE NA SPOLJNU MEMORIJU

A:#:adresa u nultoj strani
X:#:niži bajt krajnje adrese
Y:#:viši bajt krajnje adrese
S:#:*

PR :SETLFS,SETNAM
GR :5-8-9-STATUS

Rutina SAVE zahteva početnu i krajnju adresu memorijskog segmenta koji se snima. Pri tom X i Y sadrže poslednju adresu, odnosno za jedan veću, dok akumulator pokazuje na adresu u nultoj strani. U njoj se mora nalaziti niži, a u sledeći viši bajt početne adrese.

Rutine LOAD i SAVE, iako su izdvojene, ne predstavljaju nikakve posebne delove već kombinaciju ostalih rutina za slanje i primanje karaktera sa veze, pa se postiže određeni automatizam u radu. Tako se za LOAD sa SI koriste elementarne rutine LISTEN, TALK, CIOUT, ACPTR i druge. Ranije je pomenuto da je sekundarna adresa 0 rezervisana za LOAD. Ako se direktnoji diska otvori kao sekvencijalna datoteka ali upotrebi SA=0. disk će ga slati u formatu koji odgovara programu. Nema preprekе da se bilo koja programska datoteka na disku čita kao i sekvencijalna i da se dopsivanjem petlje koja će primljene znakove smeštati u memoriju, kreira sopstvena LOAD rutina.

```
        LDA #3
        LDX #<NAME
        LDY #>NAME
        JSR  SETNAM
        LDX #8
        LDY #0
        JSR  SETLFS
        LDA #0
        LDX #192
        STA  251
        STX  252
        LDA #251
        LDX #0
        LDY #208
        JSR SAVE
        ...
NAME    .ASC "MON"
```

GETIN

65508 $FFE4

UPIS KARAKTERA SA ULAZNE VEZE

```
A:#:ASCII kod karaktera, 0 ako ništa
    nije primljeno (RS232,tastatura)
X: :# samo za RS 232
Y: :# samo za RS 232
S:#:Z=1 ako A=0
```

```
PR :CHKIN
GR :STATUS
```

Rutina je prevashodno dizajnirana za upis sa tastature i RS 232. Za razliku od CHRIN, reaguje i u slučajevima kada nema podataka koje treba primiti. Ako je po povratku A=0 (Z=1), znači da ni jedan taster do tog trenutka nije pritisnut ili je ulazni bafer RS 232 uređaja prazan. Za ostale nema razlike između CHRIN i GETIN, ali „Komodor" savetuje upotrebu prve.

```
WAIT  JSR GETIN
      BEQ WAIT
      JSR SCPNT
      CMP #13
      BNE WAIT
```

CLRCHN

65484 $FFCC

ZATVARANJE VEZA ZA UPIS/ISPIS

```
A:#:-
X:#:-
Y: :-
S:#:-
```

```
PR :-
GR :-
```


Poništava dejstvo CHKIN i CHKOUT rutina. Varijable DFLTN i DFLTO ponovo dobijaju vrednosti 0 (upis sa tastature) i 3 (ispis na ekran). Ako je poslednja veza bila usmerena ka/sa SI, izvodi se UNLSN ili UNTLK.

CLALL

65511 $FFE7

ZATVARANJE SVIH VEZA

```
A:#:-
X:#:-
Y: :-
S:#:-
```

```
PR :-
GR :-
```

Bez obzira na njen naziv, ova rutina neće propisno obaviti svoj posao. Da bi sve veze bile zatvorene, bilo bi potrebno iz tablica otvorenih veza, uzimati parametre za svaku pojedinačnu i pozivati CLOSE. Ovde sa samo varijabli LOTND (152, broj otvorenih veza) dodeljuje vrednost 0 i izvodi CLRCHN! Time je operativni sistem nateran da „misli" da nema ni jedne otvorene veze, ali zatvaranje na samim periferijama nije izvedeno. Izgleda da je rutina prevashodno namenjena za zabranu pristupa vezama u nekim „kritičnim" situacijama.

CLOSE

65475 $FFC3

ZATVARANJE VEZE

```
A:#:logički broj veze
X:#:-
Y:#:-
S:#:*
```

```
PR :-
GR :0-240-STATUS
```

Ovom rutinom se poništava dejstvo OPEN. Veza sa zadatim logičkim brojem će biti izbačena iz tablica otvorenih veza i broj otvorenih veza (LDTND) smanjen za jedan. U slučaju da veza za zadatim logičkim brojem nije nikad ni otvorena, rutina neće preduzeti ništa, niti javiti grešku. Ovo je, svakako, od koristi, jer omogu-

područje mora da koristi adrese od 56296. Ostali delovi unutar izabranih 16K imaju za oba čipa iste adrese.

Ovakvo adresiranje važi ako se koriste delovi od 0 do 16K ili 32 do 48K. Ostala dva područja 16—32 i 48—64) nemaju komplikacije ovog tipa, ali zato ni mogućnost korišćenja ugrađenog generatora znakova — sve definicije oblika slova morate sami postaviti u RAM.

Kontrola nad video procesorom se uspostavlja upisivanjem odgovarajućih sadržaja u kontrolne registre. Oni su smešteni u memorijsku mapu mikroprocesora od adrese 53248, pa im se obraćamo kao i svakoj drugoj memorijskoj lokaciji. Ipak treba imati na umu da se ne radi o memoriji, jer je ponašanje nekih registra potpuno drugačije.

Sl. 3
Sl. 4

Adresa video memorije

Video memorija niske rezolucije se može nalaziti u bilo kom bloku od 1K u okviru izabranih 16 kilobajta koji su dostupni VIC. Bitovi 4—7 registra 53272 treba da sadrže početnu adresu u kilobajtima. Po uključenju je 0001 binarno, što postavlja ekran niske rezolucije od adrese 1024. Ako se video memorija premešta, treba i editoru staviti do znanja gde se nalazi promenom sistemske varijable HIBASE.

Adresa generatora znakova ili ekrana visoke rezolucije

Definicije znakova zauzimaju 2 K. Svaki od 128 znakova zahteva definiciju od 8 bajtova. Uz to se mora definisati i inverzni oblik (VIC nije u stanju sam da ga generiše), što ukupno traži navedenih 2 kilobajta. „Komodor" ima dva seta (i zato ROM od 4K), ali ako sami definišete karaktere možete kreirati samo jedan set. Definicija mora počinjati na adresi koja predstavlja paran broj kilobajta. Adresa u kilobajtima se definiše u prva 4 bita registra 53272. Pošto je parna, bit 0 je uvek nula. Ako se izabere blok 2K od 4096 ili 6144 (binarno 0100), VIC će „napraviti skok" i uzimati definicije iz ugrađenog ROM-a. Ove vrednosti se i postavljaju u inicijalizaciji računara. Pri promeni seta slova tasterima COMM+SHIFT, „komodor" jednostavno samo menja bit 1 ovog registra.

Memorija ekrana visoke rezolucije od 8K se tretira kao i generator znakova, pa joj poziciju određuju prva 4 bita registra 53272. Pošto sada adresa u kilobajtima mora biti deljiva sa 8, izbor se praktično svodi na samo dve mogućnosti: 0 što postavlja adresu 0, i 8 (binarno 1000), što daje početnu adresu 8192.

Ni svemogući VIC, međutim pak nije toliko svemoguć da na istoj adresi vidi dve stvari. Tako će ekran visoke rezolucije, ako se postavi od adrese 0, u gornjoj polovini od adrese 4096 (na ekranu monitora u donjoj) biti preklopljen generatorom znakova koji ima prioritet.

SID — Generator zvuka

Generisanje zvuka je, sigurno, jedna od prvih stvari koja se proba na novom računaru. Ako se ne računaju dugačke DATA liste sa potrebnim vrednostima za melodiju, ostalo je izgledalo dosta jednostavno: nekoliko POKE i komodor bi „zapevao" iz sve snage. Iz bejzika je to gotovo sve što se i može uraditi, ali se iz mašinskog jezika, zbog mogućnosti brze kontrole, pružaju neograničene mogućnosti.

Svaki od tri glasa SID-a kontroliše 7 registara. Jedina nevolja je što sve vremenske vrednosti (frekvencija, trajanje) zahtevaju preračunavanje iz „naših" jedinica. Tako se sadržaj registra za učestanost tona preračunava po sledećoj formuli:

SADRŽAJ REGISTRA = FREKVENCIJA (u Hz)/0.059604645

Problem se može jednostavno rešiti upotrebom tablica koje su date u uputstvu za računar. Trajanje pojedinih faza u generisanju tona koje se definišu registrima (za glas 1) 54277 i 54278 se određuju prema tablici:

VREDNOST ZA REGIST.	STVARNO VREME (u ms) ZA PORAST NIVOA	PAD NIVOA
0	2	6
1	8	24
2	16	48
3	24	72
4	38	114
5	56	168
6	68	204
7	80	240
8	100	300
9	250	750
10	500	1500
11	800	2400
12	1000	3000
13	3000	9000
14	5000	15000
15	8000	24000

Proces generisanja tona počinje upisom 1 u bit 0 kontrolnog registra 54276. Za vreme koje je definisano sadržajem poslednja četiri bita registra 54277, jačina tona će linearno rasti do maksimalnog nivoa. Zatim počinje opadanje nivoa do veličine definisane bitovima 7—4

```
49152                    .OPT P4
49152  169 000           LDA  #0
49154  141 015 212       STA  54287
49157  169 030           LDA  #30
49159  141 014 212       STA  54286
49162  169 240           LDA  #%11110000
49164  141 020 212       STA  54292
49167  141 006 212       STA  54278
49170  169 129           LDA  #129
49172  141 018 212       STA  54290
49175  169 017           LDA  #17
49177  141 004 212       STA  54276
49180  169 143           LDA  #%10001111
49182  141 024 212       STA  54296
49185  173 027 212  L1   LDA  54299
49188  141 001 212       STA  54273
49191  024               CLC
49192  144 247           BCC  L1
49194  096               RTS
```

SID

REGISTAR		BIT 7	6	5	4	3	2	1	0
54272		FREKVENCIJA niži bjt							
54273	G	FREKVENCIJA viši bajt							
54274	L	ŠIRINA IMPULSA U JEDNOM PERIODU niži bajt							
54275	A	viša 4 bita							
54276	S 1	Šum 1=da	četvrtke 1=da	testera 1=da	trougao 1=da	test 1=stop	ring modulac 1 sa 3	sinhronizacij 1 sa 3	ton 1=start 0=otpuš
54277		Vreme do maksimalnog nivoa				Vreme pada do const. nivoa			
54278		Veličina konstantnog nivoa				Vreme pada do nivoa 0			
54279		FREKVENCIJA niži bjt							
54280	G	FREKVENCIJA viši bajt							
54281	L	ŠIRINA IMPULSA U JEDNOM PERIODU niži bajt							
54282	A	viša 4 bita							
54283	S 2	Šum 1=da	četvrtke 1=da	testera 1=da	trougao 1=da	test 1=stop	ring modulac 2 sa 1	sinhronizacij 2 sa 1	ton 1=start 0=otpuš
54284		Vreme do maksimalnog nivoa				Vreme pada do const. nivoa			
54285		Veličina konstantnog nivoa				Vreme pada do nivoa 0			
54286		FREKVENCIJA niži bjt							
54287	G	FREKVENCIJA viši bajt							
54288	L	ŠIRINA IMPULSA U JEDNOM PERIODU niži bajt							
54289	A	viša 4 bita							
54290	S 3	Šum 1=da	četvrtke 1=da	testera 1=da	trougao 1=da	test 1=stop	ring modulac 3 sa 2	sinhronizacij 3 sa 2	ton 1=start 0=otpuš
54291		Vreme do maksimalnog nivoa				Vreme pada do const. nivoa			
54292		Veličina konstantnog nivoa				Vreme pada do nivoa 0			
54293		PRESEČ.F. bit 0-2							
54294		PRESEČNA FREKVENCIJA bit 3-10							
54295		REZONANCIJA FILTERA				PROPUŠTANJE GLASA KROZ FILTER 1=da spolja	glas 3	glas 2	glas 1
54296		glas 3 1=nečuj 0=čuje	TIP FILTERA visoko propus	propus opsega	nisko propus	NIVO IZLAZNOG SIGNALA			
54297		A/D KONVERTOR 1 (sa kontrol porta)							
54298		A/D KONVERTOR 2 (sa kontrol porta)							
54299		Gener.glasa 3. Ako glas 3 proizvodi šum: slučajan broj							
54300		Generator obvojnice glasa 3, prema regestrima 54291,54292							

Ovo je rutina koju poziva CHROUT kada treba da ispiše znak na ekranu. Njeno direktno pozivanje ima opravdanja, pa je šteta što „komodor" nije i njenu adresu ubacio u KERNAL tablicu. Čest je slučaj da se znaci primaju sa nekog uređaja i zatim ispisuju na štampač i ekran istovremeno. Poštujući uobičajenu proceduru, program za listanje direktorija direktno sa diska bi izgledao kao u primeru (A). Primer (B) obavlja isti zadatak, ali direktnim pozivom (u oba slučaja izostavili smo delove koji su za oba programa isti; otvaranje veze, provera kraja datoteke itd.) Razlika u dužini dva primera nije bitna. Primer A za svaki bajt mora da izvede prvo CHKIN i CHKOUT, pre ispisa i CLRCHN, dakle tri rutine od kojih svaka u sebi sadrži više elementarnih. Ukupna razlika u brzini je dovoljna da natera neke autore da koriste direktan poziv. Ovakav pristup je opravdan samo kada je kritična brzina izvršavanja programa.

```
        (A)                    (B)
DOP   LDX #10          LDX #10
      JSR  CHKIN       JSR  CHKIN
      JSR  CHRIN       LDX #4
      PHA              JSR  CHKOUT
      LDX #4     DOP   JSR  CHRIN
      JSR  CHKOUT      JSR  CHROUT
      PLA              JSR  SCPNT
      JSR  CHROUT      BCC  DOP
      PHA
      JSR  CLRCHN
      PLA
      JSR  CHROUT
      BCC  DOP
```

CHRIN

65487 $FFCF

UPIS KARAKTERA SA ULAZNE VEZE

A:#:ASCII kod upisanog karaktera
X:#:-
Y: :-
S:#:*

PR :CHKIN
GR :0-STATUS

Na osnovu DFLTN (153) variable rutina određuje sa kog uređaja treba primiti znak. Ako nijedna veza nije proglašena ulaznom, upis ide sa tastature. Pri tome se on tretira na poseban način. Pozivom CHRIN pojavljuje se kursor i sve što se otkuca ispisuje se na ekranu. Ovu rutinu napušta tek pritisak na RETURN, s tim što se kod prvog znaka nalazi u akumulatoru. Svaki sledeći poziv vraća kod sledećeg znaka sve dok i posledniji nije obrađen. Da li će izvesti upis cele linije ili vratiti kod sledećeg znaka, rutina odlučuje na osnovu variable CRSW (208). Ako je u njoj 0 izvešće se ceo upis, dok će svaka druga vratnost naterati CHRIN da samo iz video memorije uzme sledeći znak i njegov ekranski kod pretvori u ASCII. Pozicija znaka na ekranu je određena adresom početka linije PNT (209, 210) i ofsetom PNTR (211). Poziv SCNPNT vraća CRSW na nulu.

CHRIN iz video memorije je isto što i sa tastature, ali CRSW prethodno dobija vrednost različitu od nule, čime se preskače pojava kursora i upis cele linije. STATUS će označiti poslednji bajt (kraj datoteke) kada se stigne do 40. ili 80. znaka u liniji, zavisno od toga da li je linija povezana sa sledećom pri ranijem ispisu. U tom slučaju u akumulatoru se vraća bajt 13, iako se on ne može naći u video memoriji.

Uzimanje znaka iz datoteke na disku ili traci zahteva i proveru da li je uzeti bajt poslednji. Kako je u početku dato, bit 6 STATUS varijable daje ovu informaciju. Kod čitanja sekvencijalnih datoteka sa trake krajem se smatra bajt 0, pa CHRIN ispituje uvek i jedan bajt unapred. Na žalost, 0 se pri upisivanju sa CHROUT može bez ograničenja koristiti, ali će prva njena pojava pri čitanju izazvati prekid i svi podaci koji slede neće biti pročitani. Verovatno je „Komodor" smatrao da bajt nula u sekvencijalnim datotekama nema smisla.

Za RS 232 uređaje se ne preporučuje upotreba ove rutine. Ona će se vrteti u petlji dok god ne primi jedan bajt. Ako spoljni uređaj ništa i ne pošalje, može je prekinuti jedino resetovanje računara.

```
        LDY #0
GET     JSR CHRIN
        STA 512,Y
        JSR READST
        BNE EXIT
        INY
        BNE GET
```

```
BCS  ERROR
...
;**
NAME  .ASC "TEST,S,R"
```

CHKOUT

65481 $FFC9

OTVARANJE VEZE ZA ISPIS

```
A:#:-
X:#:logički broj veze
Y: :-
S:#:*

PR :OPEN
GR :0-3-5-7-STATUS
```

Da bi se neki znak poslao preko veze, potrebno je da se ona proglasi izlaznom, što je zadatak ove rutine. Nakon nje, u sistemskoj promenjivoj DFLTO (154) će se naći broj uređaja na koji je ispis usmeren. U slučaju SI, prethodno se izvode LISTEN i SECOND.

Pošto se video memorija pri inicijalizaciji proglašava otvorenom za ispis, nema potrebe otvarati vezu sa njom i izvoditi CHKOUT.

CHKIN

65478 $FFC6

OTVARANJE VEZE ZA UPIS

```
A:#:-
X:#:logički broj veze
Y: :-
S:#:*

PR :OPEN
GR :3-5-6-STATUS
```

Upis karaktera sa veze može se izvesti samo ako se veza prethodno proglasi otvorenom za upis, što je zadatak ove rutine. Nakon nje u sistemskoj promenjivoj DFLTN (153) naći će broj uređaja sa koga će biti primani karakteri. Za SI, u okviru ove rutine izvode se TALK i TKSA.

Ako ni jedna druga veza nije proglašena ulaznom, upis se dobija sa tastature, pa nema potrebe za OPEN i CHKIN

CHROUT

65490 $FFD2

SLANJE KARAKTERA NA IZLAZNU VEZU

```
A: :ASCII kod karaktera
X: :-
Y: :-
S:#:*

PR :CHKOUT
GR :0-STATUS
```

Ovo je jedna od najčešće korišćenih rutina, pa su vrednosti svih registara pri izlazu očuvane, što omogućava jednostavnu upotrebu programskih petlji. Izbor uređaja na koji će karakter biti poslat zavisi od sadržaja sistemske promenjive DFLTO (154). Ako njen sadržaj nije izmenjen u CHKOUT rutini, ispis je usmeren na ekran.

Pri slanju znakova na SI uređaje, podatke će primati svaki kome je u CHKOUT naređeno da sluša, pa je moguće slanje na više uređaja istovremeno.

```
      LDA #0
      JSR SETNAM
      LDA #33
      LDX #4
      LDY #7
      JSR OPEN
      LDX #33
      JSR CHKOUT
      LDX #20
      LDA #"*"
DOP   JSR CHROUT
      DEX
      BNE DOP
```

SCPNT

59158 $E716

ISPIS KARAKTERA NA EKRAN

```
A: :ASCII kod karaktera
X: :-
Y: :-
S:#:C=0 uvek

PR :-
GR :-
```


registra 54278. Trajanje ovog pada je vremenski kontrolisano sadržajem prva četiri bita registra 54277. Na tom nivou signal ostaje sve dok se bit kojim je proces započeo ne vrati na 0. Tada počinje linearno opadanje jačine do 0, za vreme koje je definisano registrom 54278 (bit 0—3). Potpuno analogno važi i za ostala dva glasa uz primenu adresa odgovarajućih registara.

Iako sva tri glasa izgledaju ravnopravna, ipak je glasu 3 dodeljeno nešto više funkcija. Promenom sadržaja odgovarajućih registara moguće je jedan glas modulisati drugim. Pri tome se čuje i glas kojim se vrši modulacija i modulisani glas. Ovo se može izbeći ako se koristi 3 glas kome se, u tom slučaju, sprečava izlaz na zvučnik postavljanjem 7 bita registra 54296 na 1.

Dva poslednja registra su takođe vezana za glas 3. Kada se pokrene, vrednosti trenutnog stanja amplitude mogu se pročitati u registru 54299. Ako je izabrani talasni oblik četvrtast, vrednost će se menjati sa 0 na 255 i obratno. Ako je testerast, vrednost će linearno rasti do 255 i naglo pasti na 0, pa opet rasti itd. Možda je najinteresantniji slučaj izbor šuma za glas 3. tada registar sadrži vrednost koja može odlično biti upotrebljena kao slučajan broj.

Registar 54300 u svakom trenutku pokazuje nivo signala glasa tri i tako prati oblik obvojnice koji je definisan vremenima porasta i pada nivoa.

Svaki od tri glasa se može propustiti kroz filter čija se presečna frekvencija definiše u registrima 54293 i 54294. Pri tome filter može biti niskopropusni, propusnik opsega, visokopropusni i, kombinacijom 1 i 3, nepropusnik opsega.

Posebno (potpuno neiskorišćeno zadovoljstvo) predstavljaju dva osmobitna A/D konvertora. Priključivanjem potenciometra od 470 K na odgovarajući kontakt porta za džojstik može se iz registra 54297 dobiti vrednost naměštene otpornosti (u opsegu 0—255). Brzina konverzije je oko 0,5 ms.

Programiranjem u mašinskom jeziku lako se ostvaruje veliki broj zvučnih efekata. Pri tome su posebno korisni registri 54299 i 54300 jer se njihova vrednost može stalno prebacivati u druge registre i tako postići izvanredni dinamički efekti, kao što je dato u primeru.

CIA — KOMUNIKACIJA SA OKOLINOM

Za komunikaciju sa periferijama i vremensko održavanje sistemskih funkcija u „komodoru" su zadužena dva identična CIA čipa. Kako je to već uobičajeno, samo jedan deo njihovih mogućnosti je iskorišćen, dok je ostalo prepušteno mašti korisnika. Poznavanje ovih komponenti je neophodno za realizaciju različitih hardverskih rešenja, ali postoji i nekoliko elemenata koji mogu znatno pomoći u rešavanju programskih problema.

Ulazno/izlazni port

U mikroprocesor je ugrađen jedan port kojim se kontroliše rad kasetofona i raspored memorijskih blokova. Na identičan način funkcionišu i portovi CIA, ali su iskorišćeni za druge namene. kao što je prikazano u tabeli registara.

Prva dva bita porta A. CIA 2, se takođe upotrebljavaju za kontrolu pristupa memorijskim blokovima, ali onima koje „vidi" video procesor. Blok od 16K koji će mu biti dostupan se određuje upisom sledećeg sadržaja u prva dva bita na adresi 56567:

11=0—16 K
10=16—32 K
01=32—48 K
00=48—64 K

Na istom portu se nalaze svi signali potrebni za rad serijskog interfejsa. To omogućuje da se kreira sopstveni način komunikacije sa diskjedinicom, jer joj je moguće poslati odgovarajući mašinski program koji bi ovu komunikaciju prihvatio. Za ambicioznije. jedno od mogućih rešenja se nalazi u dobro poznatom programu za brzo kopiranje disketa.

```
49152                           .OPT P4
49152  169 018                  LDA #<CINV
49154  162 192                  LDX #>CINV
49156  120                      SEI
49157  141 020 003              STA 788
49160  142 021 003              STX 789
49163  169 000                  LDA #0
49165  141 008 220              STA 56328
49168  088                      CLI
49169  096                      RTS
49170  120            CINV      SEI
49171  160 030                  LDY #30
49173  173 011 220              LDA 56331
49176  041 127                  AND #127
49178  032 052 192              JSR ASCI
49181  173 010 220              LDA 56330
49184  032 052 192              JSR ASCI
49187  173 009 220              LDA 56329
49190  032 052 192              JSR ASCI
49193  173 008 220              LDA 56328
49196  009 048                  ORA #48
49198  153 000 004              STA 1024,Y
49201  076 049 234              JMP 59953
49204  170            ASCI      TAX
49205  074                      LSR
49206  074                      LSR
49207  074                      LSR
49208  074                      LSR
49209  009 048                  ORA #48
49211  153 000 004              STA 1024,Y
49214  200                      INY
49215  138                      TXA
49216  041 015                  AND #15
49218  009 048                  ORA #48
49220  153 000 004              STA 1024,Y
49223  200                      INY
49224  169 058                  LDA #":"
49226  153 000 004              STA 1024,Y
49229  200                      INY
49230  096                      RTS
```

CIA #1

ADRESA REGISTRA	BIT 7	6	5	4	3	2	1	0
56320	TASTATURA potenciometar 0 1 =A 1 0 =B			palica za igru 2 pucanj	levo	desno	dole	gore
56321	TASTATURA izlaz broj.B	izlaz broj.A		palica za igru 1 pucanj	levo	desno	dole	gore
56322	7	SMER PRENOSA ZA PORT A (56320) (1=izlaz)						0
56323	7	SMER PRENOSA ZA PORT B (56321) (1=izlaz)						0
56324	TAJMER A niži bajt ----------(interapt OS)------							
56325	TAJMER A viši bajt							
56326	TAJMER B niži bajt							
56327	TAJMER B viši bajt ----------(serijski interfejs)--							
56328	ČASOVNIK 1/10 sec.							
56329	SEKUNDE (10)			SEKUNDE (1)				
56330	MINUTI (10)			MINUTI (1)				
56331	AM/PM		(10)	SATI (1)				
56332	BAFER SERIJSKOG PORTA							
56333	1=IRQ 1=post. 0=bris.	INTERAPT REGISTAR		kaset. SI	serijs. port	alarm časovn.	tajmer B	tajmer A
56334	frekv. za čas. 1=50Hz 0=60Hz	smer s.porta 1=izlaz 0=ulaz	tajmer A broji 1=CNT 0=1MHz	forsir. upis u tajm.A 1=da	tajm.A 1=jedan ciklus 0=kont.	izlaz na PB6 1=monos 0=clk	izlaz na PB6 1=da 0=ne	tajmer A 1=kreni 0=stani
56335	postavi 1=alarm 0=časov	tajmer B broji 00=1MHz 01=CNT 10=ciklus A 11=cik.A i CNT		TAJMER B isto kao gore za tajmer A				

Tajmer i kontrolni registar

Svakog šezdesetog dela sekunde mikroprocesor prekida svoj rad i prelazi na izvršavanje rutine za očitavanje tastature i druge sistemske funkcije. Ovaj vremenski interval obezbeduje tajmer A (CIA=1) koji neprekidno broji impulse sistemskog časovnika (unazad) i kad stigne do nule, generiše interapt i ponovo počinje sa odbrojavanjem. U jednoj CIA-i ugradena su dva ovakva tajmera, ali sa nešto različitim funkcijama. Postavljanjem odgovarajućih bitova u kontrolnom registru (56334 za CIA=1), tajmer A se može naterati da broji učestanost od 1 MHz sistemskog časovnika, ali i spoljne impulse koji se dovode na CNT izvod korisničkog porta. Promenom bita 5 i 6 kontrolnog registra 56335, tajmer B može da broji i 1 MHz i CNT, ali i broj punih ciklusa tajmera A. Tako se praktično dva brojača funkcionalno spajaju u jedan za veće periode brojanja. Moguće je i brojanje samo onih punih ciklusa tajmera A koji su se odigrali za vreme dok je na spoljnom CNT postojao signal. Na ovaj način se može meriti frekvencija spoljnjeg signala, dužina perioda i slično.

U programima se tajmeri mogu iskoristiti za pravljenje pauza ili vremensko ograničavanje. Odbrojavanje teče od trenutka upisa 1 u nulti bit kontrolnog registra i kada stigne do nule, ponovo se postavlja početna vrednost. Ako je bit 4 kontrolnog registra postavljen na 0, proces će se neprekidno ponavljati. Brojanje od neke druge vrednosti se postiže njenim upisom u sam brojač. Ako je u tom trenutku aktivan, prihvatiće novu vrednost tek po završetku tekućeg ciklusa. Moguće ga je naterati da je prihvati i odmah, bez obzira na njegovo trenutno stanje, ako se upiše 1 u bit 4 kontrolnog registra.

Registar za prekide

Kao što video procesor ima mogućnost generisanja prekida ako se odigrao neki od

Pre otvaranja datoteke na disku ili traci ovom se rutinom mora definisati njen naziv. Može se nalaziti bilo gde u memoriji — X i Y registri treba da pokazuju na njegov početak. Ako naziv nije potreban, dovoljno je pozvati rutinu sa nulom u akumulatoru.

```
SETLFS
65466                                   $FFBA
```

```
POSTAVLJANJE LOGIČKE, PRVE I SEKUNDARNE
ADRESE

A: :logička adresa
X: :prva adresa (broj uređaja)
Y: :sekundarna adresa
S: :-

PR :-
GR :-
```

Još jedna rutina potrebna za definisanje parametara OPEN, SAVE ili LOAD rutine. Logička adresa je proizvoljan broj po kojem će se veza identifikovati u daljem radu. Prve i sekundarne adrese date su u sledećoj tablici:

PRVA ADR.	UREĐAJ	SEKUNDARNA ADRESA	
0	tastatura	nema značaja	
1	kasetofon	SAVE	0 — snimanje sa zaglavljem 1 1 — snimanje sa zaglavljem 3 2 — nakon SAVE snimanje sumarke koj traka zaglavlje 5)
		LOAD	3 — kontrolacija 1 i 2 0 — uzima iz bafera 1 — upis na izvornu adresu
		OPEN	0 — čitanje datoteke 1 — upisivanje nove datoteke (zaglavlje 4) 2 — kao kod SAVE
2	RS 232	0 — unos nema značaja	
3	video memorija	nema značaja	
4	štampač	0 — velika slova/grafički znaci 7 — mala slova/velika slova ostala vrednosti zavise od tipa štampača	
8	disk	SAVE	0 — unos
		LOAD	0 — relativan upis 1 — upis na izvornu adresu
		OPEN	2 — direktan pristup 14 — otvaranje/kreiranje datoteke 15 — komandni kanal

Za SI bit 7 sekundarne adrese ima posebnu namenu pri otvaranju datoteke. Da bi se izbegle zabune u slučaju da korisnik sam definiše sekundarnu adresu veću od 127, ona neće biti poslata, pa ni ime datoteke za njom. Zato takve adrese ne treba koristiti za disk, dok je za štampač moguća 255, koji ima isti efekat kao i 0.

Brojevi zaglavlja koje smo naveli kod kasetofona odnose sa na prvi bajt u njemu. Njime se definiše tip datoteke (sekvencijalna, programska ili zaglavlje „kraj trake"). Zaglavlje 1 i 3 za operativni sistem imaju isto značenje, ali ih bejzik razlikuje. Ako je zaglavlje 3, bejzik će upisati program uvek na izvornu lokaciju bez obzira na sekundarnu adresu za LOAD.

```
OPEN
65472                                   $FFC0
```

```
OTVARANJE VEZE

A:#:-
X:#:-
Y:#:-
S:#:*

PR :SETNAM-SETLFS
GR :1-2-4-5-6-240-STATUS
```

Prebacivanje parametara u tablice logičkih, prvih i sekundarnih adresa, čime je, sa stanovišta računara, veza otvorena. Svim sekundarnim adresama se prethodno bit 6 i 5 postavljaju na 1 jer to zahteva SI — ostalima ne smeta.

RS 232 zahteva 512K memorije za ulazno-izlazne bafere. OPEN rutina će pozvati MEMTOP i od dobijene adrese na niže odvojiti potreban prostor. Kako je to u najvećem broju slučajeva i prostor za bejzik stringove, oni će time biti izgubljeni. Zato se preporučuje otvaranje veze pre definisanja alfanumeričkih promenjivih. Kada je veza otvorena, bejzik će voditi računa o smanjenom prostoru.

Adrese bafera se smeštaju u sistemske promenljive RIBUF (247, 248) i ROBUF (249, 250). Ako se u njima, pre otvaranja, nađe bilo koja adresa izvan nulte strane, gore navedena procedura će biti preskočena i zatečene adrese smatrane početkom bafera. Time se može izbeći mešanje sa bejzik podnučjem.

Otvaranje veze sa SI uređajima zahteva dodatne postupke, pa OPEN u tom slučaju poziva SDNAM rutinu.

```
LDA #8
LDX #<NAME
LDY #>NAME
JSR  SETNAM
LDA #100
LDX #8
LDY #4
JSR  SETLFS
JSR  OPEN
```

SCREEN

65517 $FFED

ČITANJE X/Y ORAGANIZACIJE EKRANA

```
A: :-
X:#:broj kolona
Y:#:broj redova
S:#:-
```

Još jedan rutina koja treba da obezbedi izvršavanje programa na različitim računarima. Svaki od njih će vratiti svoju organizaciju (C64 daje 40×25), tako da program može da bude potpuno nezavisan od stvarnog broja redova i kolona.

VEZA SA SVETOM

Sve rutine iz ove grupe za rad sa periferijskim uređajima poštuju istu konvenciju o prijavljivanju grešaka. Ako je pri povratku CARRY zastavica 1, u toku izvođenja rutine je nastala greška. Broj grešaka će se naći u akumulatoru i to sa sledećim značenjem:

A greška

0 rutina prekinuta STOP tasterom
1 previše otvorenih veza
2 veza već otvorena
3 veza nije otvorena
4 datoteka nije nađena
5 uređaj nije priključen
6 veza nije predviđena za upis
7 veza nije predviđena za ispis
8 nedostaje naziv datoteke
9 nepostojeća prva adresa
240 operativni sistem je sam odredio prostor za RS 232 bafere; ovo nije greška već upozorenje.

Ispis I/O ERROR A u trenutku nastajanja se može omogućiti rutinom SETMSG.

Ako u tablicama uz pojedine rutine pored stanja mikroprocesorskog STATUS registra (S) stoji znak „"*, znači da rutina javlja grešku na naveden način. U tom slučaju će pored ER stajati brojevi grešaka koje se mogu javiti. Ako pod ER stoji i STATUS, znači da treba ispitati i STATUS varijablu za punu informaciju o stanju komunikacije. Svaki bit STATUSA, ako je 1, ima sledeće značenje:*

bit kasetofon	*serijski interfejs*
0 —	*istek predviđenog vremena pri ispisu*
1 —	*istek predviđenog vremena pri upisu*
2 kratak blok	—
3 dugačak blok	—
4 greška u čitanju	—
5 čeksam greška	—
6 kraj datoteke	*kraj datoteke*
7 kraj treke	*uređaj nije priključen*

Istek predviđenog vremena, za serijski interfejs, odnosi se na vremenski format u kome se signali šalju. Strogo je definisano vreme za koje se pojedini signali moraju pojaviti — u suprotnom, sistem prijavljuje grešku. Jedan od slučajeva je pokušaj primanja sa uređaja koji ne šalje (nije prethodno izveden TALK).

READST

65463 $FFB7

ČITANJE STATUS VARIJABLE

```
A:#:STATUS
X: :
Y: :
S: :Z=1 ako STATUS=0

PR :-
GR :-
```

Rutina je vrlo jednostavna i svodi se, zapravo, samo na jednu instrukciju: LDA 144, međutim, za RS 232 vrednost statusa se nalazi na adresi 663, a ujedno se odmah po čitanju i briše. Zato se umesto direktnog očitavanja pomenutih lokacija uvek preporučuje poziv ove rutine. Rutine dodelju vrednost STATUSU logičkom OR operacijom, pa bitovi prethodnih grešaka ostaju. Rutine iz grupe koja sledi o tome vode računa — prethodno resetuju STATUS — ali za one u sledećem poglavlju to ne važi.

SETNAM

65469 $FFBD

POSTAVLJANJE NAZIVA DATOTEKE

```
A: :dužina
X: :početna adresa, niži bajt
Y: :početna adresa, viši bajt
S: :-

PR :-
GR :-
```


CIA #2

ADRESA REGISTRA	7	6	5	BIT 4	3	2	1	0
56576	SERIJSKI INTERFEJS					VIC mem. 16K blok		
56577	RS232 (korisnički port)							
56578	7	SMER PRENOSA ZA PORT A (56576) (1=izlaz)						0
56579	7	SMER PRENOSA ZA PORT B (56577) (1=izlaz)						0
56580	TAJMER A niži bajt							
56581	TAJMER A viši bajt							
56582	TAJMER B niži bajt							
56583	TAJMER B viši bajt -(RS232)-							
56584					ČASOVNIK 1/10 sec.			
56585		SEKUNDE (10)			SEKUNDE (1)			
56586		MINUTI (10)			MINUTI (1)			
56587	AM/PM			(10)	SATI (1)			
56588	BAFER SERIJSKOG PORTA							

56589	1=IRQ 1=post. 0=bris.	INTERAPT REGISTAR		RS232	serijs. port	alarm časovn.	tajmer B	tajmer A
56590	frekv. za čas. 1=50Hz 0=60Hz	smer s.porta 1=izlaz 0=ulaz	tajmer A broji 1=CNT 0=1MHz	forsir. upis u tajm.A 1=da	tajm.A 1=jedan ciklus 0=kont.	izlaz na PB6 1=monos 0=clk	izlaz na PB6 1=da 0=ne	tajmer A 1=kreni 0=stani
56591	postavi 1=alarm 0=časov	tajmer B broji 00=1MHz 01=CNT 10=ciklus A 11=clk.A i CNT		TAJMER B isto kao gore za tajmer A				

predviđenih slučajeva, tako i CIA ima istu funkciju, ali se nešto složenije kontroliše. VIC ima dva registra, jedan za samu detekciju događaja i drugi za dozvolu interapta od njega, dok CIA za ovu namenu ima jedan registar (zapravo dva, ali na istoj adresi). Događaji koji se registruju su: prisutnost signala na spoljnoj FLAG liniji ili upis/ispis bajta sa serijskog porta (uglavnom hardverska pitanja, pa preskačemo objašnjenja), kraj jednog ciklusa brojanja tajmera A i B, i alarm časovnika. Ako nastupi neki od događaja, odgovarajući bit interapt registra će biti postavljen na 1. Čitanjem ovog registra bit se ujedno i briše i tako priprema za ponovno registrovanje.

Da bi, međutim, interapt stvarno bio generisan, potrebno mu je za to dati dozvolu, što je nešto složenija operacija. Registar je na istoj adresi (za CIA 1 56333) i ako se želi promena nekog bita, na njegovo mesto se UVEK upisuje bit 1. Da li će time interapt biti dozvoljen ili zabranjen zavisi kako ste postavili 7 bit pri upisivanju u registar. Ako je bio 1, dozvoličete interapt, a ako je 0 interapt će biti zabranjen. Konkretno: ako želite da sprečite da tajmer A generiše interapt (da biste registre izbacili iz adresnog prostora i pročitali generator znakova, na primer) u kontroli registar se upisuje binarno 00000001. Da bi prekid ponovo bio omogućen, treba upisati 10000001. Svaki interapt od strane CIA#1 se može zabraniti, dok CIA#2 generiše nemaskirani interapt.

Tačno vreme

Ako vam je u programu potrebno tačno merenje vremena, teško da ćete se pomoći bejzik varijablom TIMERS. Ako je proces generisanja interapta tajmera A makar i jednom bio sprečen, njena vrednost gubi trku sa stvarnim vremenom. A to se dešava pri ispisu svakog bajta na disk, štampač ili kasetofon, dakle šanse za tačan rad su minimalne. RS232 će tek potpuno poremetiti stvar. Odgovor je koriš-

nje časovnika CIA koji je potpuno nezavisan od softverskih dešavanja u računaru.

Vreme je smešteno u 4 registra i to u BDC formatu koji je posebno pogodan za ispis jer nema potrebe za konvertovanjem binarnog broja u decimalni. Četiri bita daju vrednost jedne cifre od 0—9, pa se jednim bajtom može predstaviti vrednost od 00 do 99. Da bi se cifra ispisala, dovoljno je izolovati grupu od 4 bita i dodati joj 48 (ASCII kod 0). kao što je to urađeno u primeru. Pošto za minute i sekunde prva cifra može biti najviše 5, to je za nju dovoljno i tri bita.

Časovnik se pokreće na malo čudan ali efikasan način: upisom vrednosti desetinki sekunde. Zaustavlja se upisom časova. Tako se može podesiti da krene od precizno zadatog vremena. Da se pri čitanju vremena ne bi desilo da uzmemo vrednost, na primer, minuta, pa dok pročitamo sekunde vrednost bude promenjena, ugrađena je još jedna slična funkcija: kada časovnik radi, čitanje časova ce zamrznuti sve ostale vrednosti (iako časovnik i dalje neometano odbrojava), se vreme može precizno pročitati. Ovo privremeno stanje traje dok ne pročitamo registar sa desetinkama sekunde, kada registri ponovo dobijaju vrednost stvarnog vremena u tom trenutku. Dakle, sve je podešeno za maksimalnu preciznost.

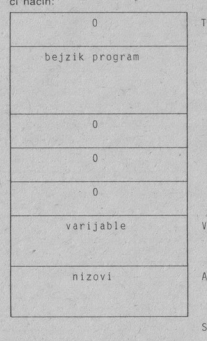
Alarm se postavlja prethodnim setovanjem bita 7 kontrolnog registra 56335. pa zatim upisom željene vrednosti u registre za vreme. Ako se preko interapt registra dozvoli, nailazak časovnika na zadato vreme će prekinuti mikroprocesor u redovnom poslu (preko vektora CINV), a ostalo zavisi od vaše maštovitosti.

Mašinac iz bejzika

Bejzik interpreterom „komodor 64" se u ovom dodatku nećemo baviti (mnogi bi rekli da tu i nema čime da se bavi) iz dva razloga. Jedan je ograničenost prostora. Drugi, važniji, je što želimo maksimalno da stimulišemo razvijanje sličnih sistemskih programa. Zato nam se činilo važnim da u ovom pregledu damo što detaljniji prikaz osnovnih komponenti i operativnog sistema kao osnove na kojoj se može dogradivati sistemski softver. Interesantna je činjenica da i komercijalni programi u pravilu slede ovaj koncept: nije nam poznat nijedan (malo bolji) program koji se oslanja na bejzik ROM. Najčešće se taj deo potpuno izbacuje iz adresnog prostora i koristi dragocenih 8 K RAM-a.

Kada je već ugrađen, pozabavićemo se njime tek toliko da se obezbedimo da nam — ne zasmeta! Pošto je najveći broj aplikacija manjeg obima, često je potpuno opravdano neki deo programa napisati i u bejziku, pa treba naći bezbedno mesto za mašinske programe.

Pri inicijalizaciji operativnog sistema određuje se najniža i najviša RAM lokacija i postavljaju sistemske promenjive MEMSTR i MEMSIZ. Kada se nakon toga inicijalizuje bejzik, ove će vrednosti biti prebačene u varijable TXTTAB i MEMSIZ) opet MEMSIZ, ali ne onaj operativnog sistema već na adresi 55—56 — izgleda da je maštovitost zakazala). Nakon toga će raspodela ovog dela memorije biti izvršena na sledeći način:



Prva neobičnost vezana za ovaj raspored je postojanje bajta 0 na adresi ispred samog programa. Promenom varijable TXTTAB možete pomeriti područje za bejzik program na neku višu adresu. Međutim, da bi se program u svim slučajevima ispravno ponašao, sadržaj lokacije T×TTAB—1 mora biti 0. Tako bi rezervisanje 2 K za mašinski program ispred bejzika zahtevalo sledeću proceduru:

```
NEW
POKE 43,1:POKE 44,16:REM početak od 4097
POKE 4096,0
NEW
```

Čini nam se da je odvajanje prostora na ovaj način potpuno zapostavljeno, iako u mnogim situacijama može biti praktično. Odvajanje prostora za definicije sprajtova obično predstavlja problem jer se korišćenjem viših adresa (ako video memorija ostaje u prvih 16K) potpuno nepotrebo ograničava prostor bejzika. Ovim načinom bejziku sa „gornje" strane ostaje sva

```
LDX #<SAVEVECT
LDY #>SAVEVECT
SEC
JSR  VECTOR
LDX #<NEWVECT
LDY #>NEWVECT
CLC
SEI
JSR  VECTOR
CLI
```

SETMSG

65424 $FF90

KONTROLA PORUKA OPERATIVNOG SISTEMA

```
A:#:bit7 = 1 ispis kontrolnih poruka
     bit6 = 1 ispis poruka o greškama
     Pri izlazu A=STATUS
X: :-
Y: :-
S: :Z=1 ako STATUS = 0
```

Rutina postavlja vrednost sistemske promenjive MSGFLG (157). Bit 7, ako je nula, sprečava ispis kontrolnih poruka u koje spadaju: PRESS PLAY ON TAPE, LOADING itd. Poruka o greškama se, u stvari, svodi samo na jednu: I/O ERROR n i verujemo da je malo koji vlasnik „komodora" imao prilike da je vidi. Njihov ispis sprečava bejzik, koji poseduje svoje, daleko jasnije poruke. Brojevi grešaka „n" su dati u okviru prikaza rutina za komunikaciju sa periferijama.

MEMTOP

65433 $FF99

ČITANJE/POSTAVLJANJE NAJVIŠE RAM ADRESE

```
A: :-
X: :niži bajt adrese
Y: :viši bajt adrese
S:#:C=1 čitanje, C=0 postavljanje
```

Najviše slobodna memorijska adresa se postavlja nakon RAM testa. Ako se koristi bejzik ROM, njena vrednost je 40960. Nakon inicijalizacije bejzika, ovom rutinom nije više moguće ograničiti prostor za bejzik programe.

MEMBOT

65436 $FF9C

ČITANJE/POSTAVLJANJE NAJNIŽE RAM ADRESE

```
A: :-
X: :niži bajt adrese
Y: :viši bajt adrese
S:#:C=1 čitanje, C=0 postavljanje
```

Važe iste napomene kao i za prethodnu rutinu. Obe dobijaju pun smisao tek prilikom dopisivanja sistemskih programa koji na sebe preuzimaju i inicijalizaciju računara. Ovim se rutinama, u tom slučaju, određuje memorijski prostor koji će biti na raspolaganju jezičkom delu.

IOBASE

65523 $FFF3

ČITANJE POČETNE ADRESE ULAZNO/IZLAZNIH JEDINICA

```
A: :-
X:#:niži bajt adrese
Y:#:viši bajt adrese
S:#:-
```

Registri CIA 1 i 2 (u prevodu: kompleksni interfejsni adapter) su smešteni u memorijsku mapu mikroprocesora i ova rutina će u X i Y registrima vratiti početnu adresu na kojoj se nalaze. Ako se, zatim, registrima pristupa indirektno indeksiranim adresiranjem, takav program će raditi na svakom „Komodorovom" računaru iz ove klase. Polazi se od pretpostavke da svi koriste iste, ili kompatibilne, komponente za komunikaciju sa periferijama i to na isti način na koji i C64. Zato bi navedeni primer trebalo na svakom računaru da spreči generisanje interapta za očitavanje tastature.

```
JSR  IOBASE
STX  20
STY  21
LDY #13
LDA #1
STA (20),Y
```

INITV

58792 $E5A8

INICIJALIZACIJA VIDEO PROCESORA (VIC II)

A:#:-
X:#:-
Y: :-
S:#:-

Video procesor poseduje 47 registara. Njihove početne vrednosti (svi sprajtovi isključeni, ekran niske rezolucije od adrese 1024, tamno i svetlo plava boja ekrana itd.) smeštene su od adrese 60601 i ovom se rutinom prebacuju u VIC registre. Može korisno da posluži za resetovanje sprajtova ili prelazak sa visoke na nisku rezoluciju.

IOINIT

65412 $FF84

INICIJALIZACIJA ULAZNO/IZLAZNIH JEDINICA (CIA 1 i 2) I ZVUKA (SID)

A:#:-
X:#:-
Y: :-
S:#:-

Portova CIA 1 čipa mikroprocesor koristi za očitavanje tastature, dok njegov tajmer A generiše maskirani interapt koji poziva rutinu za ovom namenu. CIA 2 služi kao serijski i RS232 interfejs.

Interni port mikroprocesora 6510 se postavlja za kontrolu motora i slanje podataka na kasetofon, dok stanje prve tri linije (bit 0,1 i 2) određuje izgled memorijske mape (postavljaju se sve binarne jedinice).

Jačina zvuka SID čipa se postavlja na nulu.

RAMTAS

65415 $FF87

INICIJALIZACIJA SISTEMSKOG DELA MEMORIJE, RAM TEST

A:#:-
X:#:-
Y:#:-
S:#:-

Nulta, druga i treća strana memorije se „brišu" (upisuje se 0), a 192 memorijske lokacije, od adrese 828, se rezervišu kao bafer za rad sa kasetofonom i zatim se izvodi test priključene RAM memorije. Test je nedestruktivan, pa je sadržaj RAM-a očuvan. Prva lokacija koja ne odgovori sadržajem koji je u testu u nju upisan (brojevi 85 i 170) smatra se najvišom, i vrednost MEMTOP sistemske promenljive postavlja na nju. MEMBOT (najniža adresa RAM-a) dobija vrednost 2048.

RESTORE

65418 $FF8A

INICIJALIZACIJA VEKTORA OPERATIVNOG SISTEMA

A:#:-
X:#:-
Y:#:-
S:#:-

Preko vektora u RAM memoriji poziva se 16 rutina operativnog sistema. Njihove početne vrednosti su zapisane od adrese 64816 i ovim potprogramom se prebacuju u treću stranu memorije od adrese 788. Rutina koristi VECTOR.

VECTOR

65421 $FF8D

ČITANJE/POSTAVLJANJE VEKTORA OPERATIVNOG SISTEMA

A:#:-
X:#:niži bajt adrese
Y:#:viši bajt adrese
S:#:C=0 postavljanje, C=1 čitanje

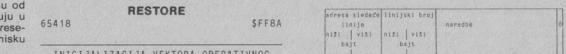
Ako je CARRY zastavica setovana pri ulasku u ovu rutinu, vrednosti vektora sa adresa 788-819 će biti prebačene u RAM, od adrese na koju pokazuju X i Y registri. Ako je C=0, dobija se obrnut proces. Pošto su i interapt rutine vektorizovane, treba prethodno izvesti instrukciju SEI. Nemaskirani interapt ovim nije spречen (generiše ga CIA 2 u radu sa RS 232 i RESTORE taster), pa o tome treba voditi računa.

inače raspoloživa memorija. Nedostatak je što se procedura mora izvesti pre upisivanja samog programa, ali se to može jednostavno rešiti prethodnim upisivanjem malog programa koji će odvojiti potreban prostor i potom upisati glavni.

```
10 POKE 43,1:POKE 44,16:POKE 4096,0
20 PRINT"{CLOAD" CHR$(34)"NNZ1V"CHR$(34)",8{RVS}
30 PRINT"RUN"
40 POKE631,19:POKE632,13:POKE633,13
50 POKE634,13:POKE198,4
60 SYS42115
READY.
```

Kraj bejzik programa

Svaka linija bejzik programa se u memoriji nalazi u formatu:



Ako su prva dva bajta (adresa sledeće linije) jednaka 0, znači da je kraj programa. Zajedno sa nulom sa kraja prethodne linije, dobijaju se ukupno 3 nule kao oznaka kraja programa. U uobičajenom radu bejzika od te adrese (VARTAB) POČINJU VARIJABLE. Ali i ne moraju. Ako se VARTAB „pomeri" na više, obrisće se slobodan prostor koji je dovoljno zaštićen, pa može koristiti za mašinski program. Jedino što može napraviti problem je ubacivanje novih linija ili izbacivanje postojećih, jer se time celo područje premešta.

Prostor na vrhu

Od najviše memorijske adrese ka kraju bejzik programa smeštaju se stringovi. Pri tome se u tom području nalaze samo sadržaji string promenjivih, dok su sam naziv dužina i početna adresa definisani kao i svaka druga varijabla u području iznad VARTAB. Možda niste znali: sve varijable u ovom delu zauzimaju 7 bajtova — dva za naziv i pet za podatak. U ovih pet se nalazi stvarna vrednost realnih promenjivih, dužina i početna adresa stringa čiji su elementi smešteni od vrha memorije i, najveće razočarenje, vrednosti celobrojnih promenjivih koje zauzimaju samo dva bajta, dok su preostala tri neiskorišćena. Ako se ovome doda činjenica da se pre svake operacije sa celobrojnim vrednostima prethono vrši konverzija u format pokretnog zareza, pa rezultat ponovo konvertuje u ceo broj, upotreba ovakvih varijabli je potpuno besmislena. Pri dimenzionisanju niza stavi stoje drugačije. Elementi niza zaista zauzimaju samo dva bajta, pa je samo u tom slučaju, ako već nemate dovoljno prostora, opravdano koristi celobrojne promenjive.

Prostor na vrhu se rezerviše na sledeći način:
CLR:POKE 55,0:POKE 56,152:CLR:REM slobodno od adrese 38912
Naredba CLR je obavezna.

Ostala mesta

Prostor od adrese 49152 predstavlja verovatno, najpopularniji izbor. Potpuno je slobodan, ne diraju ga ni bejzik ni operativni sistem, pa izgleda da je namenski i odvojen za mašinske programe. Ove deo je već stekao toliku popularnost kod onih koji rade na mašinskom jeziku da jedan, odličan, asembler podrazumeva ORG 49152 ako se drukčije ne zada.

Za ambicioznije programe postoji poseban prostor. Ako mašinski program od adrese 32767 počinje tačno definisanim zaglavljem (vidi objašnjenje rutine operativnog sistema AOINT), onda će računar podrazumevati da je on glavni izvršni program i pri svakoj inicijalizaciji (STOP+RESTORE ili SYS64738) odmah prenositi kontrolu na njega. Ovo je prvenstveno dizajnirano zbog kartridža (programa u memoriji koja se priključuje spolja), ali se može koristiti i ako su zaglavlje i program u samoj računarskoj memoriji.

RAM memorija koja se nalazi „ispod" bejzika i operativnog sistema je takođe nepravedno zapostavljena. Nije uvek neophodno da se ROM-ovi i izbacuju iz adresnog prostora. Ako program radi sa većom količinom podataka, oni se mogu smestati u navedena područja (i bez zamene ROM-RAM-a samo pri čitanju izvršiti zamenu, što je operacija koja vrlo kratko traje.

Sistemske promenjive

Mikroprocesor 6510 koji koristi računar „Komodor" ima svega tri registra, ali je dizajniran tako da efikasno koristi radnu memoriju, čime se ovaj nedostatak nadoknađuje. Zbog toga je potreban relativno veliki prostor za sistemske varijable, ali nisu sve od značaja za programere. Da bismo napravili jasan pregled, tipove podataka u sistemskom delu memorije smo podelili u sledeće grupe:

Status

U ovu grupu spadaju varijable koje ukazuju na određeno stanje računara u toku rada ili računar na osnovu njihovih vrednosti određuje kojim od nekoliko različitih puteva u programu treba da krene. Pri tome postoje dve grupe: (a) one koje se mogu menjati i tako direktno kontrolisati rad računara i (b) one koje se potpuno interno koriste, pa se ne može ostvariti nikakva kontrola promenom njihovog sadržaja. Tipičan primer za prvu grupu je promenjiva RVS(199) na osnovu koje računar određuje da li sledeći znak treba da bude ispisan inverzno. Ako joj je vrednost 0, znači će biti ispisivani normalno; dok svaka druga vrednost uslovljava ispis inverznih znakova. Kao primer za drugu grupu može da posluži C3PO (148) varijabla koju operativni sistem koristi za sinhronizaciju rada serijskog interfejsa. Njenom promenom može se jedino proizvesti — neprevidljiv efekat.

Po'am statusa koristimo dosta uslovno; nije neophodno da vrednost pojedinih bitova ili cele promenjive bude isključivo 0 ili 1. Na primer vaijabla NDX (198), ako je 0, označava da do tog trenutka ni jedan taster nije pritisut, ako je pritisnuto više tastera koji su smešteni u red i čekaju na obradu, onda njena vrednost označava koliko ih ima. Ipak čemo i ovakve varijable svrstati u svu grupu.

Pokazivači

Jedna od najvažnijih osobina mikroprocesora 6510 je mogućnost indirektnog adresiranja. U instrukciji se zada adresa neke lokacije u nultor strani,a mikroprocesor na osnovu njenog sadržaja i sadržaja sledeće formira stvarnu adresu. Pri tome se može koristiti i indeksirani oblik, kada se na ovako dobijenu adresu dodaje i sadržaj Y registra, čime se efikasno ostvaruju razne petlje za prenos i ispitivanje većeg dela memorije. Zbog toga računar nultu stranu (do adrese 255) maksimalno koristi za ovakve namene, pa se tu nalazi dosta promenjivih koje zauzimaju dva bajta i služe za ovakav način adresiranja. Za razliku od STATUS varijabli, promena pokazivača nema smisla, jer pre svakog korišćenja računar. mora da im postavi početne vrednosti, čime briše svaki prethodni sadržaj. Iste promenjive se, takođe, koriste za različite namene, pa se nečemo upuštati u njihova objašnjenja.

Vektori

Po dve memorijske adrese, posmatrane kao par, sadrže u sebi stvarnu adresu na kojoj se nalazi određena rutina koja se zato poziva indirektnim skokom preko vektora. Svrha ovakvog adresiranja je otvaranje mogućnosti za kreiranje sopstvenih rutina koje promenom odgovarajućeg vektora računar prihvata kao sopstvene. Veliki broj rutina (na primer: SAVE, LOAD, OPEN itd.) se poziva na ovaj način, pa su vektori i njihova promena posebno značajni.

Tablice

Tablice predstavljaju grupu sistemskih promenjivih iste namene i ima ih nekoliko. Iako se njihov sadržaj može slobodno menjati, verujemo da od toga ima malo koristi. Objašnjenja čemo dati u pregledu varijabli za svaku tablicu posebno.

Baferi i redovi

Deo sistemskog dela memorije je odvojen za privremeno čuvanje podataka dok ne budu obrađeni. Tako će sve što se otkuca do pritiska na RETURN biti složeno u deo memorije od adrese 512 i ten kao celina biti obrađeno. Slično se dešava i pri pritisku na neki taster. Ako je računar u tom trenutku zauzet nečim drugim, kod pritisnutog tastera će biti smešten u deo memorije koji je organizovan kao red i kasnije odatle pokupljen. Bafer može biti i samo jedna memorijska lokacija.

Brojači

Ova grupa je od najmanjeg interesa za korisnika jer se retko izmenom mogu postići korisni efekti. Računar, jednostavno, koristi memorijske lokacije da bi kontrolisao trajanje određenih internih operacija. Tako je, na primer, treptanje kursora kontrolisano sa nekoliko brojača, čijom se promenom može postići da jedan treptaj kursora bude duži ili kraći. Već sledeći će biti normalne dužine, pošto mu računar postavlja vrednost pri svakom novom ciklusu. Sličnih namena su i brojači koji se koriste za vremensku kontrolu procesa prenosa podataka na kasetofon i slično.

Interno

Određeni broj memorijskih lokacija se koristi za namene koje je teško svrstati u neku od ovih grupa. Kada, na primer, bejzik interpreter naiđe na znak navoda, počinje pretraživanje ostatka linije da bi se našao sledeći i tako odredila dužina stringa koji je ovim znacima uokviren. Pošto sličnih pretraživanja ima više, napravljeno je nekoliko rutina koje detektuju pojavu jednog od dva zadata znaka koji su definisani u varijablama CHARAC (7) i ENDCHR (8). U toku rada bejzika promena ovih promenjivih nema smisla, ali zato njihovo poznavanje može biti korisno, jer se navedene rutine mogu iskoristiti u sopstvenom programu. Za one koje se prvenstveno odnose na operativni sistem dajemo nešto kasnije potrebna objašnjenja.

*Uz adrese u pregledu stoje zvanični „Komodorovi" nazivi. Zatim sledi broj bajtova koje varijabla, bafer itd. zauzimaju, a pre objašnjenja i slovo koje označava tip promenjive prema gore navedenoj podeli. Ako objašnjenje počinje znakom, * znači da se radi o varijabli čije direktno korišćenje, po našem mišljenju, nije od interesa ili je nemoguće. Objašnjenja uz ove varijable će biti data samo radi ukupnog razumevanja i mogućnosti korišćenja lokacija u sopstvenom programu, bez posebnih detalja pogotovu što često imaju višestruku namenu, pa bi takva objašnjenja po obimu pravazišla njihov značaj. Slobodne lokacije u sistemskom delu su označene nazivom XXXXXX*

2 **XXXXXX** 1 : Slobodna lokacija.
3 **ADRAY** 2 V : Konverzija pokretni zarez--celobrojna vrednost
5 **ADRAY2** 2 V : Konverzija celobrojna vrednost — Pokretni zarez
7 **CHARAC** 1 : Znak koji se traži u bejzik liniji.
8 **ENDCHR** 1 : Alternativni karakter koji se traži u bejzik liniji.
9 **TRMPOS** 1 :*Trenutna pozicija krsora za izračunavanje sledeće TAB pozicije.
10 **VERCK** 1 S:*0=Load, 1=Verify; na osnovu ove vrednosti LOAD rutina određuje da li da izvrši upis u memoriju


```
      JMP 60983 ;>3, SERIJSKI INTERFEJS
L2    LSR
      PLA
      BCC L3
      ....
      ....        ?1, KASsuOFON
L3    ....
      ....        ?2, RS 232
```

Ovakvih ispitivanja ima mnogo u „komodoru". Možda bi bilo jednostavnije za rad da su rutine sa istom namenom za pojedine uređaje posebno izdvojene, opet sa pozivom preko tablice, pogotovo što ovakav sistem zahteva uvođenje pripremnih rutina koje će definisati koji je uređaj u tom trenutku izlazni (DFLTO) i obaviti neophodne pripremne operacije. Međutim, time bi programi postali nejasni, jer bi na osnovu adrese koja se poziva, a sada bi ih bilo daleko više, trebalo shvatiti na koji će uređaj znak biti poslat. Daleko je jednostavnije imati jednu pripremnu rutinu koja se poziva sa brojem uređaja i samo jednu rutinu za ispis.

Pravila igre

Nazivi rutina i sistemskih promenljivih su dati prema knjizi PROGRAMMER'S REFERENCE GUIDE. Nekoliko ih je, tamo gde smo to smatrali opravdanim, ipak izvađeno iz celina, pa su njihovi nazivi dati manjim slovima. Uz svaku rutinu stoji pozivna adresa u decimalnom i heksadecimalnom sistemu.

Uz spisak registara mikroprocesora dato je i kratko objašnjenje njihovih potrebnih ulaznih vrednosti ili stanja pri izlazu. Ako uz registar stoji oznaka „ ", znači da se koristi u okviru rutine i da se njegova ulazna vrednost gubi.

Registar obeležen sa S je STATUS registar mikroprocesora — skup status zastavica, od kojih je za svaku rutinu najčešće samo jedna od značaja (CARRY ili ZERO), pa ih nismo posebno odvajali.

U donjem delu tablice su pripreme rutine (obeležene PR), one koje obavezno moraju biti pozvane pre navedene. Oznaka ER se odnosi na moguće greške, o čemu su pojedinosti date u odgovarajućim poglavljima.

Sve rutine u ovom pregledu, osim prve, organizovane su kao potprogrami i pozivaju se instrukcijama JSR. Programski primeri se odnose na grupu potprograma, da bi mogle biti izvedene funkcionalne celine. Uređaji koji se priključuju preko serijskog interfejsa (disk. štampač) označeni su skraćenicom SI uređaji.

Buđenje računara

START

64738 $FCE2

INICIJALIZACIJA OPERATIVNOG SISTEMA

Glavni ulaz za potpunu inicijalizaciju računara. Rutine koje slede uglavnom se izvode u okviru potprograma START.

aoint

64770 $FD02

PROVERA AUTO-START ZAGLAVLJA

```
A:#:-
X:#:-
Y: :-
S:#:Z=1 zaglavlje postoji
```

U sklopu koncepcije jednostavnih proširenja, „Komodor" je predvideo i mogućnost da kontrolu nad računarom po uključivanju preuzme program u memoriji koja se priključuje na port za proširenja. Ona se ubacuje u adresni prostor od adrese 32768 i treba da ima zaglavlje u sledećem formatu:

32768 — adresa dela za inicijalizaciju
32770 — adresa radnog početka
32772 — sekvenca znakova CBM80 (velika slova)

Potprogram AOINT proverava da li takvo zaglavlje postoji. Ako je po izlasku ZERO zastavica 1, program START će odmah izvesti JMP (32768).

CINT

65409 $FF81

INICIJALIZACIJA EKRANSKOG EDITORA I VIDEO PROCESORA (VIC II)

```
A:#:-
X:#:-
Y:#:-
S:#:-
```

U registre video procesora postavljaju se početne vrednosti, a zatim incijalizuje video memorija niske rezolucije. Ekran se briše, postavljaju vrednosti u tablicu adresa i povezanosti ekranskih linija, brojači za treptanje kursora itd. Na kraju se, uz pomoć raster registra video procesora, određuje učestanost na kojoj radi (NTSC, PAL) i adekvatno postavlja vrednost ajmera CIA 1 čipa koji generiše interapt za očitavanje tastature.

776 **IGONE** 2 V: Upućivanje na izvršenje naredbi.
778 **IEVAL** 2 V: Obrada alfa i numeričkih izraza.
780 **SAREG** 1 R: Sadržaj za A registar mikroprocesora pre skoka na SYS rutinu. Sadržaj A registra pri povratku iz SYS.
781 **SXREG** 1 R: X registar; važi isto što i za SAREG.
782 **SYREG** 1 R: Y registar; važi isto što i za SAREG.
783 **SPREG** 1 R: STATUS registar mikroprocesora; važi isto što i za SAREG.
784 **USR** 3 V: Vektor za USR funkciju; prvi bajt je 76 (mašinska instrukcija JMP) dok sledeća dva sadrže adresu.
787 XXXXXX 1 : Slobodna lokacija.
788 **CINV** 2 V: Interapt rutina (očitavanje tastature, obrada časovnika realnog vremena itd.)
790 **CBINV** 2 V: Interapt generisan mašinskom instrukcijom BRK.
792 **NMINV** 2 V: Nemaskirani interapt (resetovanje računara, RS232).
794 **IOPEN** 2 V: OPEN (rutina operativnog sistema, kao i sve koje slede).
796 **ICLOSE** 2 V: CLOSE rutina.
798 **ICHKIN** 2 V: CHKIN rutina.
800 **ISKOUT** 2 V: CHKOUT rutina.
802 **ICLRCH** 2 V: CLRCHN rutina.
804 **IBASIN** 2 V: CHRIN rutina.
806 **IBSOUT** 2 V: CHROUT rutina.
808 **ISTOP** 2 V: STOP rutina.
810 **IGETIN** 2 V: GETIN rutina.
812 **ICLALL** 2 V: CLALL rutina.
814 **USRCMD** 2 V: Vektor koji korisnik može sam da definiše.
816 **ILOAD** 2 V: LOAD rutina.
818 **ISAVE** 2 V: SAVE rutina.
820 XXXXXX 8 : Slobodan prostor.
828 **TBUFFR** 192 R: Bafer za kasetofon; pri radu sa sekvencijanim datotekama podaci se šalju u blokovima koji se ovde formiraju; SAVE i LOAD o ovom baferu formiraju samo prvi blok, zaglavlje. sa nazivom programa.
1020 XXXXX 4 : Slobodne lokacije.

Kernal

Operativni sistem „komodora" sastavljen je od 39 funkcionalnih celina organizovanih u potprograme za obavljanje određenih elementarnih zadataka: slanje podataka na periferni uređaj, primanje podataka sa perifernog uređaja, očitavanje tastature, inicijalizacija računara itd. Očigledno da takve celine moraju da postoje u svakom računaru, bez obzira kako su izvedene, pa pisanje programa koji ih koristi ne bi trebalo da zavisi od modela. Dovoljno je za svaki znati pozivne adrese i program se može napisati bez ikakvih problema. Međutim, kada program treba da se izvodi na nekom drugom računaru, upravo te adrese predstavljaju problem. Morale bi sve biti promenjene, jer je sigurno da neće biti iste, iako se pozivaju rutine sa istim zadatkom. Još akuelniji problem predstavljaju usavršavanje operativnog sistema od strane proizvođača koja uvek rezultuju nekim manjim pomeranjem pojedinih delova.

Komplikacije koje iz ovoga proizlaze mogu se jednostavno izbeći. Dovoljno je da proizvođač na jednom mestu fiksira pozivne adrese, a odatle JUMP instrukcijama usmerava programunutar operativnog sistema na stvarne adrese koje korisnik ne mora da zna. To je „Komodor" i uradio i na poslednjoj strani memorije postavio takozvanu KERNAL JUMP tablicu, po kojoj se čeko naziva i ceo operativni sistem. Kada treba pozvati, na primer, rutinu za ispis karaktera, poziva se uvek ista adresa u tablici. Korisniku nije važno gde će program nakon toga biti usmeren JMP instrukcijom. Dok god je tablica na istom mestu stvaran raspored rutina nije bitan, a proizvođač će se postarati da za svaku promenu ili novi model računara u nju unese odgovarajuće promene.

Ove indirektne pozive „Komodor" je do maksimuma iskoristio i unutar samog sistema da bi i sebi olakšao rad na izmenama. Kada bejzik interpreter treba da ispiše jedan znak iz stringa koji ste zadali i kad se njegov kod već nađe u akumulatoru mikroprocesor mora da izvede čitav niz JMP instrukcija da bi stigao do odgovarajuće rutine:

```
43821 JSR 43847
43847 JSR 57612
57612 JMP 65490
65490 JMP (806)=JMP 61898
61898 JMP 59158
59158 ispis karaktera na ekran
```

Jasno, uz svaku ovu JMP (JSR) instrukciju postoji i mentalni deo sa određenim zadatkom. Na primer, 57612, po povratku iz JSR 65490, proverava stanje CARRY zastavice da bi ustanovio eventualnu grešku u procesu ispisa i izdao odgovarajuću bejzik poruku.

Za nas je od interesa samo jedan skok: JSR 65490. To je pozivrutine preko KERNAL tablice. Ovo je karakteristične slučaj (ukupno ih ima 10) jer koristi indiriektni skok preko vektora. Vrednost vektora 61898 se postavlja u inicijalizaciji, pa je sa stanovišta kompatibilnosti isto kao i direktan skok, ali se odbija mogućnost izmene vektora i usmeravanje programa na sopstvene rutine. Stvarna adresa 59158 na kojoj se obavlja ispis na ekran tako gubi važnost.

Slanje znakova na bilo koji uređaj postiže se pozivom iste adrese u tablici, 65490. Sistemska promenljiva DFLTO (154) sadrži broj uređaja na koji je usmeren ispis i kada pozovete rutinu za slanje (ili bilo koju drugu zaduženu za komunikaciju sa više uređaja) ona počinje sa ispitivanjem o kom se uređaju radi i vrši skok na različite delove koji će izvršiti ispis. Na primer:

```
JSR 65490
65490  JMP (806) = JMP 61898
61898  PHA
       LDA 154
       CMP #3
       BNE L1
       JMP 59158 ;3, ISPIS NA EKRAN
L1     BCC L2
       PLA
```


ili samo proveru sa sadržajem memorije; direktno menjanje nije od interesa jer se ovaj parametar postavlja kao i svi drugi ulazni parametri . LOAD.

11 **COUNT** 1 : Dužina bejzik linije koja se trenutno ubacuje u program; ulazna veličina za deo koji otvara potreban prostor, ali koji nije organizovan kao sabrutina, pa je mogućnost korišćenja ograničena.
12 **DIMFLG** 1 S:"Interni status bejzika pri dimenzionisanju niza.
13 **VALTYP** 1 S:"Tip podatka koji se obrađuje; 255=string, 0=numerik.
14 **INTFLG** 1 S:"Tip numeričkog podatka: 128=ceo broj, 0=realan broj
15 **GARBFL** 1 S:"Višenamenski radni status bejzika za GOTO, sređivanje stringova itd.
16 **SUBFLG** 1 S:"Interni status bejzika za DEF FN.
17 **INPFLG** 1 S:"Način uzimanja podatka: 0=INPUT, 64=GET, 148=READ.
18 **TANSGN** 1 S:"Znak rezultata TAN funkcije ili poređenja
19 **INPRMT** 1 S:"Da li postoji string u okviru INPUT naredbe.
20 **LINNUM** 2 :"Više namena, između ostalog akumulator za celobrojne vrednosti.
22 **TEMPPT** 1 P:"Privremeni stek za string operacije.
23 **LASTPT** 2 P:"Poslednja korišćena adresa za privremeno držanje stringa.
25 **TEMPST** 9 R:"Prostor za string operacije.
34 **INDEX** 4 P:"Višenamenski prostor, najčešće kao dva pokazivača za prenos stringova, ispis i slično.
38 **RESHO** 5 R:"Rezultat množenja u formatu pokretnog zareza.
43 **TXTTAB** 2 P:Početak bejzik teksta.
45 **VARTAB** 2 P:Početak bejzik varijabili.)47 **ARYTAB** 2 P:Početak bejzik nizova.
49 **STREND** 2 P:Poslednja adresa (+1) koju zauzimaju bejzik nizovi.
51 **FRETOP** 2 P:Najniža adresa koju zauzimaju stringovi.
53 **FRESPC** 2 P:"Adresa stringa koji se trenutno obrađuje.
55 **MEMSIZ** 2 P: Najviša memorijska adresa koju koristi bejzik.
57 **CURLIN** 2 : Broj bejzik linije koja se trenutno izvodi.
59 **OLDLIN** 2 : Broj prethodne bejzik linije.
61 **OLDTXT** 2 P: Adresa naredbe u okviru bejzik programa od koje će CONT nastaviti rad.
63 **DATLIN** 2 : Linijski broj DATA naredbe sa koje se uzima sledeći podatak.
65 **DATPTR** 2 P : Adresa podatka u okviru DATA linije koji će biti pročitan sa READ.
67 **INPPTR** 2 P :"Pokazivač za operacije u okviru READ i INPUT naredba.
69 **VARNAM** 2 : Naziv bejzik varijable koja se obrađuje.
71 **VARPNT** 2 P: "Adresa samog podatka koji se dodeljuje varijabli.
73 **FORPNT** 2 P: "Adresa FOR varijable u bejzik programu.
75 ------ 22 R: "Radni prostor za numeričke operacije
97 **FACEXP** 1 R : Numerički akumulator #1: eksponent.
1:eksponent.
98 **FACHO** 4 R : Numerički akumulator#1: mantisa.
102 **FACSGN** 1 R : Numerički akumulator#1: znak.
103 **SGNFLG** 1 P : "Interno u numeričkoj obradi.
104 **BITS** 1 R : "Numerički akumulator#1: indikator prekoračenja maksimalne vrednosti brojeva.
105 **ARGEXP** 1 R: Numerički aku ulator #2: eksponent.
106 **ARGHO** 4 R: Numerički akumulator #2: mantisa.
110 **ARGSGN** 1 R: Numerički akumulator #2: znak.
111 **ARISGN** 1 R: Znak rezultata poređenja NAK#1 i NAK#2.
112 **FACOV** 1 R: "Interno u numeričkoj obradi.
113 **FBUFPT** 2 P: "Višenamenski: numerička obrada, rad sa kasetofonom.
115 **CHRGET** 24: POTPROGRAM: uzimanje bajta iz bejzik teksta. Adresa bajta se menja u samoj instrukciji mikroprocesora LDAxxxx koja se nalazi na adresi 121; zato se sadržaj lokacija 122, 123 koristi i kao pokazivač; koristi se i ulaz u rutinu na adresi 121 što kao rezultat daje isti bajt koji je prethodno dobijen sa CHRGOT 115
139 **RNDX** 5 R: Vrednost za izračunavanje sledećeg slučajnog broja (format: klizni zarez)
144 **STATUS** 1 S: STATUS varijabla operativnog sistema (u bejzicu ST). Informacija o stanju komunikacije sa svim perifernim uređajima sem RS232.
145 **STKEY** 1 S: 127=pr i poslednjem ispitivanju taster STOP pritisnut.
146 **SVXT** 1 B: "Vremenska kontrola upisa/ispisa sa kasetofona.
147 **VERCK** 1 S: "0=LOAD, 1=VERIFY. Isto kao na adresi 10.
148 **148 C3PO** 1 S: "Postoji znak u BSOUR koji čeka na ispis.
149 **BSOUR** 1 R: "Znak koji se šalje preko serijskog interfejsa.
150 **SYNO** 1 R: "Sinhronizacija za u/i sa kasetofona.
151 ------ 1 R: "Privremeno za u/i sa kasetofona.
152 **LDTND** 1 P: Broj otvorenih veza sa OPEN.
153 **DFLTN** 1 : Broj uređaja sa koga se trenutno vrši upis.

154 **DFLTO** 1 : Broj uređaja na koji se trenutno vrši ispis, ili je trajno usmeren naredbom CMD.

155 **PRTY** 1 R: *Interno za u/i sa kasetofona.

156 **DPSW** 1 S: *Primljen bajt sa kasetofona.

157 **MSGFLG** 1 S: Ispis poruka operativnog sistema: bit7=1 ispis kontrolnih poruka, bit 6=1 ispis poruka o greškama.

158 **PRT1/2** 2 S: *Interno za upis sa kasetofona.

160 **TIME** 3 R: Časovnik „realnog vremena".

163 _____ 2 : *Privremeni smeštaj za podatke.

165 **CNTDN** 1 B: *Vremenska kontrola za kasetofon.

166 **BUFPNT** 1 P: *Preskok u baferu za kasetofon.

167 **INBIT** 1 R: *Interno RS232/kasetofon.

168 **BITCI** 1 R: *Interno RS232/kasetofon.

169 **RINONE** 1 S: *RS232: Provera start bita.

170 **RIDATA** 1 *RS232: Primljeni karakter.

171 **RIPRTY** 1 R: *RS232: Stanje parity bita.

172 **SAL** 2 P: *Privremeno za skrolovanje ekrana i kaset. bafer.

174 **EAL** 2 P: Adresa poslednjeg bajta se šalje sa SAVE.

176 **CMPO** 2 B: *Interno kasetofon.

178 **TAPE1** 2 P: Početak bafera za u/i sa kasetofona; postavlja se samo u inicijalizaciji računara pa je promenom TAPE1 moguće koristiti bilo koji deo memorije za kas. bafer.

180 **BITTS** 1 B: *RS232: brojač poslatih bitova.

181 **NXTBIT** 1 S: *RS232: sledeći bit za slanje. kasetofon: primljena oznaka „kraj trake".

182 **RODATA** 1 R: *RS232: Karakter koji se šalje.

183 **FNLEN** 1 : Dužina naziva datoteke za OPEN.

184 **LA** 1 : Logički broj veze preko koje je izvršena poslednja komunikacija.

185 **SA** 1 : Sekundarna adresa vezana za LA.

186 **FA** 1 : Prva adresa (broj uređaja) vezana za LA.

187 **FNADR** 2 : Početna adresa naziva datoteke za OPEN.

189 **ROPRTY** 1 : *RS232 *Parity bit.

190 **FSBLK** 1 B: *Brojač ispisanih/upisanih blokova sa kasete.

191 **MYCH** 1 R: *Bajt koji se šalje preko serijskog interfejsa.

192 **CAS1** 1 S: Kontrola kaset. motora. U okviru IRQ rutina je ugrađena logika koja na osnovu stanja ove varijable i tastera na kasetofonu uključuje ili isključuje motor.

193 **STAL** 2 P: *Privremeni pokazivač za rad sa periferním uređajima.

195 **MEMUSS** 2 P: *Privremeni pokazivač za određivanje početne adrese za LOAD.

197 **LSTX** 1 R: Broj tastera koji je trenutno pritisnut; postavlja se pri svakom IRQ; vrednosti od 0—64 (64=nijedan taster nije pritisnut) služe za određivanje ASCII koda.

198 **NDX** 1 S: Broj ASCII kodova u redu za tastaturu (KEYD) koji čekaju na obradu.

199 **RVS** 1 S: Ako je na ekran u inverznom obliku, 0=normalan ispis; direktna promena vrednosti ove varijable je identična ispisu kontrolnih karaktera 18 i 146.

200 **INDX** 1 : Logička dužina linije upisane naredbom INPUT (ili rutinom CHRIN za tastaturu); ako je upis počeo u prvoj koloni, ujedno i apsolutna dužina upisanog stringa.

201 **LXSP** 2 : *XY pozicija kursora na početku INPUT-a.

203 **SFDX** 1 S: *Pri određivanju ASCII koda.

204 **BLNSW** 1 S: Kontrola kursora; ako je vrednost 0, kursor se pojavljuje na ekranu bez obzira što se ne očekuje upis; svaka druga vrednost sprečava pojavu kursora.

205 **BLNCT** 1 B: *Brojač za trajanje jednog treptaja kursora.

206 **GDBLN** 1 R: Karakter koji se trenutno nalazi „ispod" kursora.

207 **BLNON** 1 S: Status kursora u jednom ciklusu treptaja: 0=kursor se ne vidi, 1=kursor se vidi.

208 **CRSW** 1 S: INPUT ili GET sa tastature; ako je 0, bio je izvršen upis cele linije (INPUT, do pritiska na RETURN) i vrednost promenjena u CRSW>0; ako je >0 biće upisan samo jedan karakter (GET); šira objašnjenja su data uz rutinu CHRIN za slučaj upisa sa tastature.

209 **PNT** 2 P: Apsolutna adresa ekranske linije na kojoj je trenutno PRINT pozicija.

211 **PNTR** 1 P: Preskok (ofset) od PNT do trenutne PRINT pozicije na liniji; može biti i veći od 40 ako su linije logički povezane.

212 **QTSW** 1 S: Indikator ispisa neparnog broja znakova navoda; ako je QTSW<>0 editor se ponaša kao da je ispisan neparan broj znakova navoda pa svi kontrolni karakteri (sem 13 i 141 i 20) gube funkciju.

213 **LNMX** 1 : Fizička dužina ekranske linije.

214 **TBLX** 1 P: Fizički broj linije na kojoj je kursor.

215 _____ 1 : *Pomoćna varijabla pri prevođenju ekranskog koda u ASCII.

216 **INSRT** 1 S: Indikator upotrebe INS tastera; ako je INS taster upotrebljen za otvaranje slobodnih mesta u liniji, INSTR služi kao brojač ubačenih karaktera; dok je vrednost veća od 0 editor je u takozvanom INSERT modu pa kontrolni karakteri (sem 13 i 14) gube funkciju.

217 **LDTB1** 26 T: Tablica povezanosti ekranskih linija; sadrži više bajtova apsolutnih adresa svake linije sa bit 7=1; ako je linija logički povezana sa prethodnom bit 7=0.

243 **USER** 2 P: Adresa PRINT pozicije u kolor RAM-u.

245 **KEYTAB** 2 P: *Pokazivač na jednu od četiri tablice pri prevođenju LSTX u ASCII kod, u zavisnosti od stanja SHIFT, COMM i CTRL tastera.

247 **RIBUF** 2 P: RS232: Početna adresa ulaznog bafera.

249 **ROBUF** 2 P: RS232: Početna adresa izlaznog bafera.

251 **FREKZP** 4 : Slobodne lokacije; vrlo se često koriste u programima jer su uvek sigurno slobodne lokacije u nultoj strani.

255 **BASZPT** 1 : *Koristi bejzik pri izvođenju STR$ funkcije.

256 **STACK** 255 : Stek mikroprocesora (od 511 na niže), stek za GOSUB, FOR . . . NEXT, radni prostor za STR$ funkciju (od 256 na više).

512 **BUF** 89 R: INPUT bafer. INPUT sa bilo kog uređaja privremeno smešta karaktere u ovaj prostor dok ne detektuje kod 13, kada se string prosleduje na obradu; u direktnom modu ujedno i radni prostor za tokenizaciju naredbi itd.

601 **LAT** 10 T: Tablica logičkih adresa svih veza koje su u tom trenutku otvorene.

611 **FAT** 10 T: Tablica prvih adresa (brojevi uređaja) vezanih za LAT.

621 **SAT** 10 T: Tablica sekundarnih adresa vezanih za LAT.

631 **KEYD** 10 R: Bafer za tastaturu; organizovan kao red u kome se čuvaju kodovi tastera koji čekaju na obradu.

641 **MEMSTR** 2 P: Najniža memorijska adresa; postavlja se u inicijalizaciji operativnog sistema i na osnovu nje bejzik pri svojoj inicijalizaciji određuje najnižu RAM lokaciju koju može da koristi.

643 **MEMSIZ** 2 P: Najviša memorijska lokacija; ista namena kao MEMSTR. .

645 **TIMOUT** 1 S: *Sprečava vremensku kontrolu odziva perifernog uređaja u slučaju da se koristi IEEE interfejs kartica; u osnovnoj konfiguraciji računara nije od značaja.

646 **COLOR** 1 R: Broj boje kojom se vrši ispis.

647 **GICOL** 1 R: Boja pozadine „ispod" kursora.

648 **HIBASE** 1 P: Strana na kojoj počinje video memorija niske rezolucije (viši bajt apsolutne adrese); ovo je vrednost koju koristi editor i mora se usaglasiti sa adresom koja je zadata video-procesoru (VIC) ako se želi promena.

649 **XMAX** 1 S: Maksimalna dužina bafera za tastaturu (<=10)

650 **RPTFLG** 1 S: Kontrola automatskog ponavljanja pritisnutog tastera; ako je bit 7=1, svi testeri imaju mogućnost auto-ponavljanja; ako je bit 6=1, svi testeri gube ovu osobinu; ako je RPTFLG=0, auto-ponavljanje poseduju samo testeri za kontrolu kursora i razmaknica.

651 **KOUNT** 1 B: *Brojač za brzinu auto-ponavljanja.

652 **DELAY** 1 B: *Brojač za razmak između dva ponavljanja.

653 **SHFLAG** 1 S: Status SHIFT, COMM i CTRL tastera; bit 0=1: pritisnut SHIFT, bit 1=1:, pritisnut COMM, bit 2=1; pritisnut CTRL.

654 **LSTSHF** 1 S: *Prethodno stanje SHFLAG.

655 **KEYLOG** 2 V: Vektor za rutinu koja na osnovu SHFLAG postavlja pokazivač KEYTAB. Nije potprogram, pa ako se vektor menja, izlazak iz nove rutine mora biti sa JMP 60128.

657 **MODE** 1 S: bit 7=1: zabrana promene seta karaktera upotrebom COMM+SHIFT tastera; postavlja se ispisom kontrolnih karaktera 8 i 9 ali direktna promena MODE ima potpuno isti efekat.

658 **AUTODN** 1 S: *Koristi editor kao indikaciju potrebe skrolovanja ekrana.

659 **M51CTR** 1 R: RS232: Kontrolni registar.

660 **M51CDR** 1 R: RS232: Komandni registar.

661 **M51AJB** 2 R: RS232: Nestandardna brzina prenosa (ako se takva zada u okviru OPEN).

663 **RSSTAT** 1 S: RS232: STATUS varijabla. Isto kao STATUS (141) ali za RS232.

664 **BITNUM** 1 B: *RS232: Broj bitova preostao za slanje.

665 **BAUDOF** 2 R: RS232: Brzina prenosa jednog bita u mikrosekundama (praktično broj otkucaja sistemskog časovnika mikroprocesora).

667 **RIDBE** 1 P: RS232: Preskok za trenutno poslednji karakter u ulaznom baferu.

668 **RIDBS** 1 P: RS232: Preskok za trenutno prvi karakter u ulaznom baferu.

669 **RODBS** 1 P: RS232: Preskok za trenutno prvi karakter u izlaznom baferu.

670 **RODBE** 1 P: RS232: Preskok za trenutno poslednji karakter u izlaznom baferu

671 **IRQTMP** 2 R: Privremeni smeštaj IRQ vektora (CINV) za vreme komunikacije sa kasetofonom; kasetofon zahteva posebnu IRQ rutinu pa se CINV vektor . menja; na ovaj način se obezbeduje da eventualna izmena od strane korisnika standardnog CINV vektora bude sačuvana.

673 **ENABL** 1 S: *RS232: Vrednost potrebna za generisanje kontrolnog bajta za registar CIA čipa kojim se startuje/zaustavlja prenos podataka; ujedno sinhronizacija RS232/serijski interfejs.

674 _____ 4 : *Privremeni radni prostor za komunikaciju sa kasetofonom.

678 **PALNTC** 1 S: *Indikator sistema 0=NTSC, 1=PAL.

679 **XXXXXX** 89 : Slobodne lokacije.

768 **IERROR** 2 V: Bejzik rutina za ispis poruka o greškama.

770 **IMAIN** 2 V: Radni početak bejzika.

772 **ICRNCH** 2 V: Tokenizacija bejzik linija.

774 **IQPLOP** 2 V: Ispis službenih reči u okviru LIST.

biblioteka

programa

Spektrum

Makazama po ROM-u

Ovaj tekst je namenjen najviše onim vlasnicima „spektruma" koji nemaju mogućnosti ili nameru da kupuju disk ili mikrodrajve, a ipak bi hteli da poboljšaju upotrebljivost svog računara pri pisanju programa i zabavi. Prekrajanje ROM-a danas, kada jedan eprom od 16 K nije skuplji od 1500 dinara (15 din), predstavlja, bez sumnje, najjeftiniju zabavu. Izmene u ROM-u koje slede treba više shvatiti kao predlog i za razmišljanje — one su ograničene jedino zvanjem i maštom programera.

Većina vlasnika „spektruma" je, verovatno, primetila da kod „spektruma" bez mikrodrajvova postoje četiri neiskorišćene naredbe. To su MOVE, ERASE, CAT i FORMAT. Na njihovo se mesto dosta jednostavno mogu ubaciti nove naredbe.

Glavni parametri svih bejzik komandi se nalaze u tabeli koja počinje od adrese &1A7A. Tu je za svaku naredbu naznačena njena klasa, potrebni separatori i izvršna adresa. Na primer, klasa 0 znači da uz naredbu ne idu nikakvi operandi. Takve naredbe su NEW, STOP CLS itd. Klasa 3 znači da uz naredbu može slediti numerički izraz. Takve naredbe su RUN i CLEAR. Klasa 6 znači da uz naredbu mora slediti numerički izraz — tu spada, na primer, GO TO. Klasa 8 označava da iza naredbe moraju slediti dva numerička izraza, razdvojena zarezom — tu spadaju POKE i OUT.

Za izmenu naredbe je potrebno na odgovarajuće mesto u tabeli upisati klasu naredbe i njenu izvršnu adresu. Tabela sa imenima naredbi počinje na adresi &0095 — treba paziti da se ne promeni ukupan broj i redosled tokena, inače će računar ispisivati sasvim druge naredbe.

Umesto nekorišćenih naredbi, autor je ubacio sledeće naredbe: QSAVE (quick save), QLOAD (quick load) i FRE (ispisuje slobodnu memoriju za bejzik) dok je četvrto mesto, zasad, slobodno.

Izmene u tabeli tokena:
STARO:
& 011F 43 41 D4 46 4F 52 4D 41 D4 4D 4F 56 C5 45 52 41 53 C5 — „CAT" „FORMAT", „MOVE", „ERASE"
NOVO:
& 011F 46 52 C5 46 4F 52 4D C1 51 53 41 56 C5 51 4C 4F C4 — „FRE", „FORMA", „QSAVE", „QLOAD"

Izmene u tabeli parametara naredbi:
STARO:
& 1B0A 0A 2C 0A 00 (MOVE)
&1B10 0A 00 93 (FRASE)
&1B14 00 93 17 (CAT)
NOVO:
& 1B0A 08 00 6E 38 (QSAVE)
& 1B10 03 96 38 (QLOAD)
& 1B14 00 BE 38 (FRE)

Za one koji dosta koriste mašinski jezik će dobro doći naredne izmene, koje omogućavaju upis i ispis u heksadecimalnom formatu. Za upis pre broja treba staviti znak „&". Broj mora biti ceo, pozitivan i najviše sa četiri cifre, mogu se upisivati i mala i velika slova (A i F), dok će se u listingu pojaviti uvek velika slova. Za ispis treba pre brojnog izraza staviti znak „%" (procenat), to jedino ima smisla kod naredbe PRINT. I ovde važi ograničenje od četiri cifre. Znak „%" može stajati ispred varijable ili numeričke formule.

Upis:
STARO:
& 2684 CD 88 2C CALL &2088
NOVO:
& 2684 CD EC 38 CALL &38EC

Ispis:
STARO:
& 202C CD FB 24 CALL &24FB
NOVO:
& 202C CD 29 39 CALL &3929

Da bi sve ove naredbe i funkcije i radile, treba još uneti i njihove izvršne rutine. One se nalaze u listingu 1 i treba ih ukucati u GENS ili neki drugi assembler i asemblirati (naravno relocirano). Potprogrami za SAVE i LOAD su, zbog štednje prostora, dati u nex-listi. Treba ih uneti tako da u EPROM-u počinju od adrese &3974. Inače, ovaj sistem ubrzanog snimanja je oko 2 i po puta brži od normalnog, što je, po nekim iskustvima, i najveća brzina pri kojoj je učitavanje još dovoljno pouzdano. Radi jedinstvenosti, moguće je snimati samo blok memorije. Zaglavlje sadrži samo 4 bajta koji označavaju početak i dužinu bloka. Pri QSA-

LISTING 1

```
10
20 ;     IZVRŠNE RUTINE
30
40      ORG #386E
50
60 ;     QSAVE COMMAND
70      CALL #1E99   ;Smesti dužinu
80      PUSH BC      ;bloka u BC.
90      CALL #1E99   ;Adresa početka
100     PUSH BC      ;bloka.
110     POP IX
120     POP DE
130     CALL QSAVE2
140     CALL #053F   ;Testiranje BREAK
150     RET          ;tastera i vraća-
160                  ;nje boje okvira.
170 QSAVE2 PUSH DE
180     PUSH IX
190     LD IX,0
200     ADD IX,SP
210     LD DE,4
220     CALL QSB     ;Header.
230     POP IX       ;Početak bloka.
240     POP DE       ;Dužina bloka.
250     CALL QSB
260     RET
270
280 ;     QLOAD COMMAND
290     CALL #1E99   ;Smesti početnu
300     PUSH HL      ;adresu u BC.
310     PUSH HL      ;Napravi mesta za
320     PUSH BC      ;učit. headera.
330 QLHL LD IX,2
340     ADD IX,SP
350     LD DE,4
360     CALL QLB     ;Učitavanje hea-
370     JR NC,QLHL   ;dera.
380     POP BC
390     POP IX
400     POP DE
410     LD A,B       ;Ukoliko nije zada-
420     OR C         ;ta adresa upisa,
430     JR Z,SBA     ;uzima se adresa
440     PUSH BC      ;iz headera.
450     POP IX
460 SBA CALL QLB
470     CALL #053F
480     RET C
490     RST 8        ;Tape loading
500     DEFB 26      ;error.
510
520 ;      FRE COMMAND
530     LD A,2       ;Otvaranje kanala
540     CALL #1601   ;za ispis gore.
550     XOR A
560     LD DE,MFRE-1
570     CALL #0C0A   ;Ispis teksta.
580     CALL #1F1A   ;Rutina za izračun.
590     LD HL;0      ;slobodne memorije.
600     OR A
610     SBC HL,BC
620     LD B,H
630     LD C,L
640     CALL #2D2B   ;Smesti sadržaj BC
650             ;na kalkulatorski stek.
660     CALL #2DE3   ;Ispis zadnje vred-
670              ;nosti sa kalk. steka.
680     LD A,13
690     RST #10
700     RET
710 MFRE DEFB 13     ;Nov red.
720     DEFM "Free memory"
730     DEFB ":"+#80
740
750 ;     HEX UPIS
760     CP "&"
770     JP NZ,#2C88
780     POP HL       ;Odbaci RET-adresu.
790     CALL #2530   ;Proveri, da li je
800        ;u toku izvršavamje prog-
810        ;rama ili provera sintakse.
820     JP NZ,#2685 ;Ovo je RUN-time.
830     LD DE,0
840     LD B,4       ;Max. 4 cifre.
850 HIL RST #20      ;Uzmi sledeći znak.
860     SUB #30      ;Razlika između
870      ;ASCII koda i br. vrednosti.
880     JR C,HE
890     CP 10  .
900     JR C,DEC
910     SUB 7        ;Plus razlika kod
920                  ;slova.
930     RES 5,A      ;Pretvori mala
940                  ;slova u velika.
950     CP 16        ;Slova mogu biti
960     JR NC,HE     ;samo od A do F.
970     CP 10
980     JR C,HE
990     RES 5,(HL)   ;U listingu će sva
1000                 ;slova biti velika.
1010 DEC EX DE,HL
1020    ADD HL,HL    ;Množenje sa 16,
1030    ADD HL,HL    ;u stvari pomeranje
1040    ADD HL,HL    ;cifre za jedan
1050    ADD HL,HL    ;ulevo.
1060    LD E,A
1070    LD D,0       ;Uključivanje nove
1080    ADD HL,DE    ;cifre.
1090    EX DE,HL     ;(RST #20 kvari HL)
1100    DJNZ HIL
1110    RST #20
1120 HE  LD B,D
1130     LD C,E
1140     CALL #2D2B  ;Smesti broj na
1150     JP #2695    ;kalkulat. stek.
1160
1170 ;     HEX ISPIS
1180     RST #18
1190     CP "%"
1200     JP NZ,#24FB
1210     POP AF          ;Odbaci RET-adresu.
1220     RST #20         ;Uzmi sledeći znak.
1230     CALL #24FB  ;Rutina za izraču-
1240         ;navanje rezultata formule.
1250     BIT 6,(IY+1) ;Ako je rezultat
1260     JP Z,#1C8A   ;string, skok.
1270     CALL #1FC3   ;Sprečava ispis
1280               ;vrednosti sa steka
1290        ;prilikom provere sintakse.
1300     CALL #1E99  ;Smesti vredn. u BC.
1310     LD A,"&"    ;Prefix za hex. na-
1320     RST #10     ;čin ispisivanja.
1330     LD D,4      ;4 cifre najviše.
1340     CALL PB
1350     LD B,C
1360     CALL PB
1370     RET
1380 PB  LD A,#F0
1390     AND B
1400     RRA
1410     RRA
1420     RRA
1430     RRA
1440     CALL PS
1450     LD A,#F
1460     AND B
1470     JP PS
1480 PS  DEC D
1490     JR Z,MPR
1500     OR A
1510     RET Z       ;Vodeća nula.
1520 MPR LD D,1
1530     ADD A,#30   ;Razlika između
1540     CP #3A      ;vrednosti i ASCII
1550     JR C,NUM    ;koda.
1560     ADD A,7     ;Dodatna razlika
1570 NUM RST #10     ;kod slova.
1580     RET
1590 QSB EQU #3974
1600 QLB EQU #39D3
```

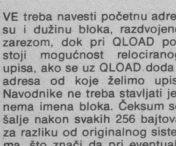
LISTING 2

```
10
20        NEMASKIRANI INTERAPT
30
40      ORG #0066
50      LD SP,(#5CB2)   ;RAMTOP.
60      DEC SP
70      LD HL,#1303     ;Adresa rutine
80      PUSH HL         ;prekida.
90      LD A,#3F
100     JP #04AA        ;Neiskorišćen
110                     ;prostor u ROMu.
120     ORG #04AA
130     LD I,A
140     LD (#5C3D),SP   ;Error SP.
150     LD IY,#5C3A     ;IY mora nositi
160     LD (IY+49),2    ;ovu vrednost u
170     IM 1            ;Basicu.
180     EI
190     RST 8           ;BREAK-CONT
200     DEFB 12         ;repeats.
```

LISTING 3

```
10 ;
20 ;         START/NEW
30 ;
40     ORG #11CB
50     LD C,A
60     LD A,7
70     OUT (#FE),A
80     LD A,#3F
90     LD I,A
100    DEFB 0,0,0,0,0,0
110    LD B:#D0       ;Briše se samo
120    XOR A          ;područje
130    LD HL,#5C00    ;sistemskih
140 CSV  LD (HL),A    ;varijabli.
150     INC HL
160     DJNZ CSV
170     EXX
180     LD (#5CB4),BC  ;Vrati P-RAMT,
190     LD (#5C38),DE  ;RASP/PIP &
200     LD (#5C7B),HL  ;UDG-samo kod NEW.
210     EXX
220     INC C
230     JR Z,#1219    ;NEW.
240     LD (#5CB4),DE   ;Fizički RAMTOP
250                   ;na #FFFF.
260     LD  HL,#FF58    ;Adresa za UDG,
270     LD (#5C7B),HL   ;ali slova se ne
280                ;kopiraju iz ROM-a
290     LD HL,#5DBF     ;RAMTOP na 23999.
300     JR #1212
```

```
QUICK SAVE/LOAD

     Najpre otkucajte hex-loader
i startujte ga-a onda, red po red,
unesite masinski program. ENTER
pritiscate tek na kraju svakoga
reda tj. posto otkucate devet
bajtova. Osim od njih ce biti
upitani u memoriju dok je deventi
zaduzen za proveru koreknosti
kucanja.

  10 REM CTRL CODE HEX LOADER

  20 BORDER 1: PAPER 1: INK 7: P
OKE 23624,0
  30 CLEAR 25499: POKE 23658,0:
PRINT AT 20,0
  40 INPUT "START ADDR:";A
  50 INPUT (A);"; ";LINE A$: IF
A$="" THEN STOP
  60 LET C=0: IF LEN A$<>18 THEN
 GO TO 90
  70 FOR I=1 TO 17 STEP 2: LET X
=16*CODE A$(I)-7*(A$(I)>"9")+
"):+CODE A$(I+1)-48-7*(A$(I+1)>"9
"): IF (C/17<>X) THEN POKE A,X: L
=A+1: LET C=C+X
  80 NEXT I: LET A=A-8: IF C-25E
+INT (C/256)*X THEN PRINT #1;"
!A$: LET A=A+8: GO TO 50
  90 PRINT #1;" FLASH 1;"ERROR!"
BEEP 1,1: GO TO 50


CTRL code hex list:

47476: F30E012100A06A00D5
47484: CDC3392BAFBC20F877
47492: 062DCD8370063SCD09
47500: C839DD6E007CAD67DC
47508: CD8139DD233E7FDB4F
47516: FE1FD01B7BB720EA44
47524: 6CCDB1397AB720E256
47532: 063B10FEC937C9E5E5
47540: C83B0406141802063E
47548: 37CDC339AF18EFCD83
47556: C6399E3C57910FE13
47564: D3FEC1EE0C4FC9F397
47572: 3E0CD3FEDBFEE6401A
47580: 4FBFC9CD3B3AA3AF3A
47588: 212002F0FE2B7CB5AD
47596: 20F9CD373A3E8B6678
47604: 02CD373A50E478FE4A
47612: 6130E02420F1064DD9
47620: CD3B5A50D53E32B06F
47628: 30F4CD3B5AD9260F5C
47636: 0633CD523AD60DD75B4
47644: 007CAD67DD23067320
47652: 1B7BB720EDCD523AB5
47660: D07CADC006333BA20CC
47668: 1137C9CD3B5ADA79A
47676: 05C8DBFEA9E640289D
47684: F7779F4F7BA9E607FF

47692: F600D3FE37C92E01FE
47700: CD375AD07BFE1ECB6D
47708: 1506333E3C9075B9C
```



VE treba navesti početnu adresu i dužinu bloka, razdvojene zarezom, dok pri QLOAD postoji mogućnost relociranog upisa, ako se uz QLOAD doda i adresa od koje želimo upis. Navodnike ne treba stavljati jer nema imena bloka. Čeksum se šalje nakon svakih 256 bajtova za razliku od originalnog sistema, što znači da pri eventualnom pogrešnom upisu nekog bita na početku bloka greška neće biti detektovana tek na kraju učitavanja, već praktično istog momenta.

Za ovako ubrzano snimanje su, ipak, potrebni kvalitetniji kasetofon i dobre kasete.

Ovim rutinama je popunjeno oko 500 bajtova u praznom delu ROM-a, koji, inače, počinje na adresi &386E i dugačak je 1170 bajtova. Kompatibilnost sa komercijalnim programima nije narušena.

Na listingu 2 data je rutina koja omogućuje nemaskirani interapt na „spektrumu". Pritiskom na NMI taster, koji treba naknadno ugraditi, moguć je uspešan povratak u bejzik (sa porukom Break-cont repeats) u 99 slučajeva, kao što su razna blokiranja, zaštićene igre itd. Jedino ako dođe do promene određenih sistemskih promenljivih, povratak neće biti uspešan. Za takve tvrdokorne slučajeve predviđena je izmena, data u listingu 3, koja menja način inicijalizacije računara pri restartu. Nema provere ni brisanja memorije, već se fizički RAM-TOP postavlja na 65535, RAM-TOP na 23999, a UDG karakter set se ne kopira na kraj memorije. Ovim je postignuto da celokupno područje od 24000 do 65535 ostane nedirnuto pri resetovanju. Naravno, treba ugraditi i reset taster, ako ga vec nemate. Tehnika pripreme ROM-a za programiranje opisana je u tekstu „ROM od sedam milja" (Računari 6 i 7).

Za ugradivanje eproma u „spectrum" postoji više načina. Može se staviti na mesto originalnog ROM-a, ali tada treba invertovati signal na nožici 27 — za to može da posluži i jedna od neiskorišćenih kapija u IC 74 (LS 00). Moguće je eprom priključiti preko konektora na zadnjoj strani, mada je sam konektor gotovo skuplji od eproma. Možda je najbolje eprom priključiti paralelno postojećem ROM-u i preklopnikom birati koji ROM ćemo koristiti. Način vezivanja dat je na priloženoj šemi. Umesto mehaničkog prekidača, može se, kao u tekstu „ROM od sedam milja" upotrebiti i elektronski, gde bi se izbor ROM-a vršio OUT instrukcijom.

Petar Putnik

Spektrum

Ključ za sve brave

U Računarima, a i u drugim časopisima, bio je spominjan uređaj pomoću kog se može bilo koji (zaštićen) program iz „spektruma" snimiti na mikrodrajv ili traku na taj način da se ceo RAM posalje na medium za snimanje. Ovo se može postići i bez nabavke tog uređaja iz Engleske ubacivanjem odgovarajuće rutine u „spektrumov" ROM. O načinu izmene ROM-a možete pročitati u članku „Poboljšanje spektrumovog ROM-a".

Da bi se sačuvalo kompletno stanje „spectruma", potrebno je na traku snimiti ceo RAM, sadržaj svih registara procesora, kao i statuse interapt flip-floopova IFF 1 i IFF 2. Sa snimanjem memorije nema problema, dok se snimanje registara obavlja tako što se njihov sadržaj pomoću PUSH instrukcije polaze na mašinski stek. Prilikom učitavanja, koje se obavlja komandom PRINT USR 15000, prvo se učitava SP (stek mora biti na istom mestu kao i pri SAVE), a zatim kompletan RAM. Nakon uspešnog učitavanja, iz steka se „vade" sadržaj svih registara i adresa prekida i program se izvršava dalje kao da i nije bilo prekida.

U listingu je korišćen sistem ubrzanog snimanja, čije su rutine za SAVE/LOAD bajtova date u tekstu „Makazama po ROM-u". Ukoliko neko želi da snimanje obavlja standardnim „spektrumovim" rutinama, treba da na odgovarajućim mestima u listingu ubaci potrebne izmene. Treba dodati LD A, FF i SCF (ovo samo kod LOAD), a i pozive subrutina treba tako obaviti da se preskoče naredbe LD HL, &053F i PUSH HL. inače će biti problema sa interaptom.

Ukoliko dođe do greške prilikom učitavanja, neće biti ispisano „Tape loading error" već „Break-cont refeats" preko rutine za povratak u bejzik. Ovo je uvedeno zbog toga da bi se izbegao eventualni krah programa različitih RAMTOP-ova prilikom SAVE i LOAD.

Da bi čitav sistem funkcionisao, potrebno je u „spektrum" ugraditi taster i vezati ga između mase i nožice NMI procesora, a na adresu 0066 treba staviti naredbu JP&3B0!. Pritiskom na NMI taster prekida se izvršavanje programa i na borderu se pojavljuju linije kao indikator. Tada imamo tri mogućnosti: pritisnuti taster „C" (continue) nastavlja se izvršavanje programa, pritiskom na dugme „B" se vraćamo u bejzik, a pritiskom na „M" se šalje cela memorija na traku (pre toga, naravno, treba startovati kasetofon).

Postoji mogućnost da se NMI ulaz aktivira i prilikom otpuštanja tastera i tako nakon učitavanja nećemo imati autostart. Da bi se to izbeglo, taster ne treba direktno vezati na nožicu NMI, već preko kondenzatora od oko 10 nF, sa kojim treba paralelno vezati otpor od oko 100K.

Glavna korist od ove rutine nije razbijanje zaštita (snimak se može učitati samo „spectrumom" koji ima iste rutine na istoj adresi), već mogućnost snimanja bilo koje trenutne pozicije na traku kod kompleksnijih igara, kao i mogućnost pauze. Ko želi, može dodati i opciju za snimanje ekrana na traku.

Petar Putnik

```
10
20 ;    UČITAVANJE CELE MEMORIJE
30 ;       I STANJA PROCESORA
40
50    ORG 15000
60 LAB LD IX,&4000
70    LD DE,2
80    CALL &LB
90    JR C,HOK
100   EI
110   RST 8
120   DEFB 26  ;Tame loading error.
130 HOK LD HL,(&4000)
140   LD A,H
150   OR L          ;SP ne sme imati
160   JR Z,LAB      ;vrednost 0.
170   LD SP,(&4000)
180   LD IX,&4000
190   LD DE,&C000
200   CALL &LB
210 RESTR JR NC,BASIC ;U slučaju greške
220 ;kod učitavanja skok na NMI rutinu.
230   PUSH IY
240   POP IX
250   POP HL
260   POP DE
270   POP BC
280   EXX
290   POP HL
300   POP DE
310   POP BC
320   EX AF,AF'
330   POP AF        ;Stanje I registra
340   LD I,A        ;i IFF-a.
350   JP P,D,DISABL
360   EI
370 DISABL CP &3F
380   IM 1
390   JR Z,MODE1
400   IM 2
410 MODE1 POP AF
420   RET           ;Skok u program.
430
440 ;    NMI-BASIC
450 BASIC LD SP,(&5CB2)  ;RAMTOP.
460   DEC SP
470   LD HL,#1383   ;Adresa rutine
480   PUSH HL       ;prekida.
490   LD A,&3F
500   JP &0&&0      ;Neiskorišćen
510                 ;prostor u ROMu.
520   ORG #04AA
530   LD A
540   LD (&5CB0),SP  ;Error SP.
550   LD IY,&5C3A    ;IY mora nositi
560   LD (IY+49),2   ;tzvu vrednost u
570   IM 1           ;Basicu.
580   EI
590   RST 8          ;BREAK-CONT
600   DEFB 12        ;repeats.
610
620 ;      NMI RUTINA
630   ORG #3AFF
640   DEFB #FF       ;Ovo je zbog kompatibil-
650   DEFB #FF       ;nosti sa nekim igrama.
660   PUSH AF
670 BOEFF LD A,6
680   OUT (#FE),A
690   LD A,#FE
700   IN A,(#FE)     ;Taster "C".
710   BIT 3,A
720   JR NZ,KT
730   POP AF
740   RETN           ;Povratak u program.
750 KT LD A,#7F
760   IN A,(#FE)
770   BIT 4,A        ;Taster "B".
780   JP Z,&0&&0
790   BIT 2,A        ;Taster "M".
800   LD A,1
810   OUT (#FE),A
820   JR NZ,BOEFF
830 ;SNIMANJE CELE MEMORIJE I STANJA
840 ;      PROCESORA NA TRAKU
850   LD A,1         ;Transfer I registra
860                  ;i IFF2-a u AF.
870   PUSH AF
880   EX AF,AF'
890   PUSH AF
900   PUSH BC
910   PUSH DE
920   PUSH HL
930   EXX
940   PUSH BC
950   PUSH DE
960   PUSH HL
970   PUSH IX
980   PUSH IY
990   LD HL,(&4000)
1000  EXX
1010  LD (&4000),SP
1020  LD IX,&4000
1030  LD DE,2        ;Header, koji nosi
1040  CALL &5EB      ;samo vrednost SP.
1050  EXX
1060  LD (&4000),HL
1070  LD IX,&4000
1080  LD DE,&C000
1090  LD HL,RESTR    ;Smesti na stek
1100                 ;adresu rutine za vracanje
1110  PUSH HL  ;sadrzaja registara.
1120  CALL &5EB
1130  SCF
1140  RET  ;  ;Nastavak preko RESTR.
1150 &5EB EQU #3974
1160 &LB EQU #3903
```


Spektrum

Microdrive Loader

Pored katastrofalne sintakse, rad sa mikrodrajvom ima

još jedan problem: katalog naredbu. Pošto smo sintaksu već sredili u „računarima“ 6 („Sintaksni teror“), na red je došao katalog. Pre svega, možemo da koristimo neki program koji će poboljšati CAT naredbu, ali to nije ni suviše praktično, ni uvek izvodljivo.

Nema zato druge nego da usvojimo konvenciju o imenu programa. Neka prvo slovo označava vrstu programa, zatim sledi jedan razmak, i na kraju, samo ime programa, za šta ostaje osam znakova. Prvo slovo može da bude:

A (AUTO RUN PROGRAM) — bejzik program snimljen sa LINE.
P (PROGRAM) — običan bejzik program.
C (CODE) — mašinski program ili blok memorije.
F (FILE) — datoteka.
X — razno
Evo nekih primera:
A BETA
C gens3m
P VLADA
F OBRAČUN

Ako sada zadamo CAT naredbu, bar ćemo znati šta je šta. Pošto smo to tako fino rešili, ostaje da vidimo učitavanje. Kada uključimo mašinu, zaista je vrlo neprijatno kucati gomilu zvezdica, navodnika i tačka-zareza da bi se učitao BETA BASIC, DEVPAC, COMPILER ili neki drugi program. To rešava ovaj mikrodrajv loader.

Zgodno je imati loader na svakoj trakici, i to pod imenom „run“. Otkucamo, dakle, RUN, pritisnemo ENTER, učita se loader i na ekranu se pojavi:

```
        MICRODRIVE LOADER

C=CAT/X=CLEAR/ENTER=ERASE
S=STOP/MDF=MDRV. FACILITY

PROGRAM IDENT: [A] P OR C

SELECT:
```

Loader očekuje da otkucamo ime programa koji treba učitati. To može da bude AUTO RUN program (A), običan bejzik program (P), ili mašinski program (C). Primetite da u slučaju AUTO RUN programa možemo, ali i ne moramo, da kucamo A razmak na početku imena. AUTO RUN program će biti učitan na mesto loadera i odmah startovan. Učitavanje običnog bejzik programa se razlikuje utoliko što se završava porukom „NONSENSE IN BASIC“, ali to je normalno — ne obraćajte pažnju. Najzad, u slučaju mašinskog programa, posle učitavanja loader će biti izbrisan iza

```
1 BORDER 0: PAPER 0: INK 7: POKE 23658,8: CLEAR : INPUT INVERSE 1;"    MICRODRIVE LOADER
"' INVERSE 0'"C=CAT/X=CLEAR/ENTER=ERASE"'"S=STOP/MDF=MDRV. FACILITY"''"PROGRAM IDENT: [A] P OR C"''"SELECT:"; LINE A$: IF A$="C" THEN CAT 1: PAUSE 0: GO TO 1
2 IF A$="" OR A$="S" THEN GO TO 9
3 IF A$="MDF" THEN GO TO 8
4 IF A$(1)="X" AND A$(2)<>" " THEN CLEAR VAL A$(2 TO ): PRINT #0; INVERSE 1;"CLEAR": PAUSE 20: GO TO 1
5 IF A$( TO 2)="P " THEN LOAD *"M";1;A$: STOP
6 IF A$( TO 2)<>"C " THEN LOAD *"M";1;"A "+A$
7 LOAD *"M";1;A$CODE : GO TO 9
8 CLEAR 65204: RESTORE 8: FOR I=65205 TO 65364: READ X: POKE I,X: NEXT I: RANDOMIZE USR 65205: DATA 33,1,0,205,110,25,229,33,10,0,205,110,25,209,205,229,25,62,254,237,71,237,94,207,255,0,62,63,237,71,237,86,201,42,243,254,126,50,8,92,35,34,243,254,58,242,254,61,50,242,254,58,59,92,203,239,50,59,92,24,94,0,0,0,42,34,77,34,59,49,59,34,34,8,1,255,245,229,58,59,92,245,255,241,203,111,32,67,58,242,254,254,0,32,194,58,8,92,254,208,40,35,254,209,40,31,254,210,40,27,254,213,40,30,254,214,40,26,254,239,40,22,254,248,40,18,254,207,32,25,62,1,33,250,254,24,12,62,9,33,246,254,24,5,62,10,33,245,254,50,242,254,34,243,254,225,241,251,237,77
9 IF A$<>"S" THEN RESTORE 9: FOR I=23296 TO 23315: READ X: POKE I,X: NEXT I: RANDOMIZE USR 23296: DATA 33,1,0,205,110,25,229,33,10,0,205,110,25,209,205,229,25,195,176,27
```

memorije kao da ga nikad nije ne bilo — za to je zadužena linija 9.

Umesto imena programa, možemo da otkucamo jednu od pet komandi koje stoje na raspolaganju; C daje katalog kertridža; X spušta RAMTOP — vrlo korisno pre učitavanja mašinskog programa — X40000 na primer, spušta RAMTOP na 40000; ENTER briše loader iz memorije; S zaustavlja program. Najzad, vrlo zanimljiva naredba MDF (MICRODRIVE FACILITY) spušta RAMTOP na 65204, zatim počev od 65205 smešta 160 bajta dugačak mašinac, aktivira ga, i na kraju briše loader. Posle toga, ako otkucamo neku naredbu poput SAVE, LOAD, ERASE, FORMAT, itd, automatski će biti ispisane sve proklete Sinclerove zvezdice, tačkice i kukice. Program, dakle, radi isto što i SYNTAX TERROR iz „Računara 6“. Potpuno je providan za korisnika i samo zauzima 160 bajta na vrhu memorije.

Dosta teorije. Da vidimo kako to radi u praksi. Pretpostavimo da treba učitati BETA BASIC (čiji je prvi deo snimljen na trakicu pod nazivom „A BETA“). Uključimo mašinu i kucamo RUN. Na pitanje „SELECT:“ odgovaramo sa BETA. I to je sve — posle par sekundi BETA BASIC će biti učitan i spreman za upotrebu.

Neki korisnici velikih zahteva će možda poželeti da prošire mogućnosti ovog loadera, pa na kraju nije loše baciti jedan pogled na sam program. Linije od 1 do 7 su jasne same po sebi. Linija 8 predstavlja MDF opciju, a linija 9 je zadužena za brisanje loadera iz memorije. To se radi tako što se u printer bafer upiše 20 bajta dugačak mašinac za tu svrhu, a zatim sa RANDOMISE USR 23296 aktivira. Primetite da opcija MDF posle aktiviranja sa RANDOMISE USR 65205 u liniji 8 takođe briše loader, ali bez petljanja sa printer baferom. Najzad, ako dodate još koju liniju programu, prepravite deveti podatak datoteke u liniji 8 i takođe deveti podatak datoteke u liniju 9 tako da sadrži broj poslednje linije u programu+1.

Vladimir Kostić

„Galaksija plus“

Ttrodimenzionalna grafika

```
10 !
20 ! TRODIMENZIONALNA GRAFIKA
30 !
40 !    D. RISTANOVIC 1985.
50 !
60 !     "RACUNARI BROJ 8"
70 !
80 HOME
90 PRINT "UNESI FUNKCIJU"
100 INPUT X$
110 GRAPH
120 Y=0
130 X=0
140 PLOT 16*Y,69+(Y+2+VAL(PTR X$))*6.23
150 DRAW 16*(X+Y),69+(Y-X+2+VAL(PTR X$))*6.23
160 X=X+0.25:IF X<8.1 GOTO 150
170 DRAW 16*(Y+8),69+(Y-X+2)*6.23
180 Y=Y+0.25:IF Y<8 GOTO 130
190 X=0
200 PLOT 16*X,69-6.23*X+10
210 Y=0
220 DRAW 16*(X+Y),69+(Y-X+2+VAL(PTR X$))*6.23
230 Y=Y+0.25:IF Y<8.1 GOTO 220
240 X=X+0.12:IF X<8 GOTO 200
250 A=KEY (0)
260 GOTO 80
```

Kratak bejzik program koji ovde dajemo je odgovoran za snimke ekrana objavljene uz prikaz „galaksije plus“: uz njegovu pomoć možete da crtate mreže svih funkcija dve promenljive koje mogu da se iskažu sa Z=f(X,Y).

Pošto startujete program sa RUN, računar će zatražiti da unesete funkciju koristeći standardno bejzik označavanje; ukoliko, dakle, želite da crtate grafik funkcije *z=sin (x*y)*, otkucaćete *SIN (X*Y)* i pritisnuti *ENTER*. Obzirom da se definicija funkcije pamti u X$, maksimalna dužina teksta je 16 znakova!

Mana programa je što je izuzetno spor, ali tome nema pomoći; ne bismo ga mnogo ubrzali ni da smo ga napisali na asembleru! Ukoliko vas ne interesuje sam proces crtanja, možete da isključite interapt i tako do konačne slike dođete četiri puta brže!

Dejan Ristanović

Komodor 64

Funkcijski tasteri

Iako „komodor 64“ poseduje 4 odvojena funkcijska tastera, njihova upotreba nije dokumentovana. Pri pritisku određenog tastera registruje se njegov kod, ali pri dekodiranju tastature se ništa ne preduzima. Kodovi funkcijskih tastera:

f1 133
f3 134
f5 135
f6 136

Istovremenim pritiskom SHIFT i funkcijskog tastera raspoznaju se:

f2 (F1+SHIFT) 137
f4 (f3+SHIFT) 138
f6 (f5+SHIFT) 139
f8 (f6+SHIFT) 140

Pomoću GET naredbe moguće je očitati funkcijske tastere i to:

```
10 GET aS:IF a$"" THEN 10
20a=ASC(a$)
30 IF a=133 THEN 1000:REM f1
40 IF a=134 THEN 1200:REM f3
```

Naravno, ovakav način korišćenja tastera je vrlo nepraktičan (proverite!) jer ne omogućava upotrebu funkcijskih tastera nezavisno od ukucanog programa — nije moguće programirati LIST i druge funkcije bez unošenja potpunog haosa u program. Priložena rutina će „promeniti“ rutinu za dekodiranje tastature, dodavši joj deo koji služi za raspoznavanje funkcijskih tastera. Pri tome je omogućeno detektovanje ne samo 8 već 16 novih funkcija, jer program ispituje da li je pored tastera pritisnut CTRL, SHIFT ili COMMODORE+aster.

Redosled tastera se, doduše, menja tako da se mora koristiti sledeća tabela:

pritisnuti taster		novo stanje
f1	=>	f1
f3	=>	f2
f5	=>	f3
f7	=>	f4
f1+SHIFT	=>	f5
f3+SHIFT	=>	f6
f5+SHIFT	=>	f7
f7+SHIFT	=>	f8
f1+COMMODORE	=>	f9
f3+COMMODORE	=>	f10
f5+COMMODORE	=>	f11
f7+COMMODORE	=>	f12
f7+COMMODORE	=>	f12
f1+CTRL	=>	f13
f3+CTRL	=>	f14
f5+CTRL	=>	f15
f7+CTRL	=>	f16

Tasteru se može dodeliti proizvoljan tekst maksimalne dužine 10 slova (ograničenje zbog veličine bafera za tastaturu), s tim što se mogu i postaviti kontrolni karakteri (za RETURN, boje) i sl. Prilikom kucanja programa u linijama 320 i 310 su već postavljene određene naredbe i to po redosledu: f1, f2, f3, f4 . . . . . Naravno, sadržaj DATA linija možete promeniti, ali prilikom izmena vodite računa o sledećem: ako želite da posle naredbe sledi RETURN, na kraju reči postavite ← Na primer LIST ← će obezbediti automatsko izvršenje posle pritiska tastera. Ako želite da koristite ze

```
10 REM ********************************
20 REM FUNKCIJSKI TASTERI
30 REM U PROGRAMU SE DEFINISU FUNKC.
40 REM TASTERI PO ZELJI KORISNIKA
50 REM ********************************
60 REM
70 REM          BOGDAN PETROVIC
80 REM
90 REM ********************************
100 PRINT"FUNKCIJSKI TASTERI-SACEKAJTE !"
110 FOR I=0 TO 86
120 READ X:POKE I+49152,X:IS=S+X:NEXT
130 DATA 169,11,160,192,141,142,2,140,144,2,96,164
140 DATA 203,196,197,240,10,177,245,201,137,176,4,200
150 DATA133,176,3,76,76,235,233,133,133,197,10,10
160 DATA 101,197,10,174,141,2,224,1,240,14,224,2
170 DATA 240,7,224,4,208,9,24,165,40,24,105,40
180 DATA 24,105,40,170,160,0,189,0,193,240,9,153
190 DATA 119,2,232,200,192,10,208,242,132,198,162,255
200 DATA 76,39,235
210 IF S<>10706 THEN PRINT"GRESKA U DATA LINIJAMA":END
220 AD= 12*4096+256
230 FOR I = 0 TO 15
240 READ X:
250 FOR J = 1 TO LEN (X$)
260 A=ASC(MID$(X$,J,1)):IF A=95 THEN A=13
270 IF A=39 THEN A=34
280 POKEAD+10*I+J-1,A:NEXT
290 IF J<11 THEN POKE AD+10*I+J-1,0
300 NEXT
310 DATA LIST+,RUN+,GOTO,CHR$(,"RENUM",SAVE,PRINT.THEN
320 DATAPRINTA,INPUTA,"LOADA",,"NEXT,GOSUB,RETURN,CLOSE,"OPEN1,8,15"
330 PRINT"OK-PROGRAM SE AKTIVIRA SA SYS49152"
340 PRINT"IZMENOM DATA LINIJA SA NAREDBAMA MOZETE"
350 PRINT"DEFINISATI TASTERE PO ZELJI"
360 END
READY.
```

ke navoda, nijh obavezno zamenite sa ' . Primer PRINT'. Ukoliko su vam potrebni, znake navoda otvorite pre reči a zatvorite posle kao „LOAD '8' ,8" ili „OPEN1,8,15".

Početak programa se nalazi na adresi $C000 ili 49152 dekadno. Počev od $C100 smeštaju se podaci o tasterima. To znači da se mašinski deo programa neće izbrisati NEW naredbom, tj. da nema bojazni od kucanja NEW (naravno, samo ako ste program propisno otkucali i startovali sa RUN, pa posle SYS 49152-posle toga slobodno pritiskajte funkcijske tastere). Program možete snimiti na kasetu ili disk u izvornom obliku — iako ćete malo čekati da se unošenje izvrši, ali, za uzvrat. možete slobodno podešavati neve funkcije ili samo „izvršnu" proceduru. U tom slučaju, program se aktivira samo sa SYS49152 i praktično trenutno je na raspolaganju, ali više niste u prilici da vršite nikakve izmene. Izvršnu proceduru snimite na sledeći način:

```
POKE 44, 192:POKE 43.0
POKE 46,194:POKE  45.0
POKE 56,208
SAVE „FUNKC TASTERI" ,8 .1 za disk a za kasetu
SAVE „FUNKC TASTERI" ,1,1 ili za kasetu sa TURBO TAPE-om.
—„FUNKC TASTERI" ,1,1
```

Napomena: posle pritiskanja STOP i RESTORE, program će se deaktivirati, ali će se sa SYS 49152 ponovo startovati.

Bogdan Petrović

mali oglasi

SPEKTRUM

• **Spektrum, prodaja i razmena programa. Besplatan spisak 600 fabričkih programa. Miroslav Radosavljević, Braće Nedića 2/VI, 11000 Beograd, 011/436-•137 od 7—15 sati**

• **Spectrum hardware. Prodajem: eprom 27128 (4500 d); komplet čipova (4164 i drugi) za proširenje RAM-a 16K na 48 ili 80K (1100 d); Komplet delova za „Megarom". Takođe osigurava kvalitetan i brzo (3 dana) servis „spektruma" i proširenje RAM. N. Četković, I. Leskovara 1, 42000 Varaždin, tel. 042/38-56 (19—21h)**

• **Programi za „spektrum", kompleti, pojedinačno, literatura, hardware modem, smanjeno grejanje, sound booster, kompjuterski pozivač telefona! Sve vrlo jeftino! Katalog! Computer studio, tel. 011/625-833.**

NEW DATA

SPECTRUM — AMSTRAD — SCHNEIDER
Programi za stručnjake.
STATISET 01, 02 i 03.
LINSET — transportno i mrežno programiranje.
GEODET — 12 programa za geodete.
Posebno za Amstrad:
CHECKER — program za kontrolu ispravnosti tastature, joysticka i kasetofona.
CHECKER + PROTON + DATA KASETA + PTT = 750 dinara!
NEW DATA, D. Brašovana 8/10, 21000 Novi Sad

• Spectrum: Ubedljivo najjeftiniji paketi na YU tržištu. Zahtevajte nov besplatan katalog sa preko 1000 programa. Radulović Rade, Vojvođanski Pot 10, 61000 Ljubljana, tel. 061/225-588.

SPEKTRUM
P.N.P. electronic
HARDVER ZA SPECTRUM
POVOLJNO PRODAJEMO KEMPSTON JEDNOSTRUKE I DVOSTRUKE INTERFACE ZA PALICE, DOMAĆE I STRANE PALICE, I/O INTERFACE, A/D I D/A KONVERTORE, AUDIO POJAČALA, SVIJETLOSNE OLOVKE, MEGA ROM, EPROM PROGRAMATOR TE PREPRAVLJENI ROM ZA SPECTRUMA. POPRAVLJAMO SPECTRUME I COMMODORE. PROGRAMIRAMO EPROME. BESPLATAN KATALOG. PNP electronic Jeretova 12 Split

Komodor

C O M M O D O R E 64
Veliki izbor literature i najkvalitetnijih programa
— štampane skripte — stručno prevedena literatura
— UPUTE uz programe
— sipmski i namjenski programi, nove igre
Kvalitetno snimljeni programi uz GARANCIJU i prma vašem izboru!!!
BESPLATAN katalog s opisima svakog programa.
NENAD RADOSAVLJEVIĆ, Cvijete Zuzorić 39/6, 41000 ZAGREB

„COMMODORE 64 — Programi, literatura i uputstva. Početnici do hakera obratite se za popis (besplatan) ili katalog (200 din) sa popisom svih programa. Sve je odlično i jeftino! Diana Krastić, Višnjan, Zagrebačka 12"

RAZNO

• Atari 600XL/800XL/400/800/130XE. Veliki izbor Atari software. Igre, korisni programi, literatura. Besplatan katalog. Lacmanović Dejan. Sindelićeva 31-A, Zrenjanin, tel. 023/31-665.

• Prodajem sofware za amstrad CPC 464. Tražite spisak programa. Kotur Rajko, Adamičeva 9, 61117 Ljubljana, 061/578-957.

• Programer daje časove programiranja i pravi programe po narudžbini za sve kućne i profesionalne računare. Tel. 777-421

• HITOVI IGRE ZA C-64! NAJBOLJI FABRIČKI PROGRAMI ZA MALO NOVCA. UVERITE SE U BRZINU I KVALITET ISPORUKE. TRAŽITE BESPLATAN KATALOG, M&S SOFTWARE, III BULEVAR 130/193, 11000 N. Beograd, Telefon: 011-146-744

• Prodajem najkvalitetniji izbor sofvera i literaturu za AMSTRAD CPC—464. Za svaku informaciju se obratite na adresu: PALM computer Florićeva 14, 41040 ZAGREB

• Prodajem AMSTRAD PRINTER INTERFACE. Jedinstven interface za pretvaranje paralelnog (CENTRONIKS) izlaza u serijski (RS—232). Okvirna cijena 13000 d. PALM computer Florićeva 14, 41040 ZAGREB

LITERATURA ZA MIKROKOMPJUTERE — KOMPLETNO PROFESIONALNO PREVEDENA I ŠTAMPANA:
COMMODORE 64 — PROGRAMER'S REFERENCE GUIDE (500 str.) — 1350 D, BASIC PRIRUČNIK —670D, SIMON'S BASIC — 670 D, PASCAL — 480 D, ZVUK I GRAFIKA NA C—64 — 780 D, UMJETNOST GRAFIKE — 950 D
AMSTRAD CPC 464 (SCHNEIDER) — UPUTSTVO ZA RAD NA AMSTRADU — 1400 D, LOCOMOTIVE BASIC — 1300 D, ARHITEKTURA I OPERATIVNI SISTEM CPC 464 — 1800 D, GRAFIKA I ZVUK ZA CPC 464 — 850 D
SPECTRUM — BASIC PROGRAMIRANJE I BROŠURA „UVOD" — 700 D
ISPORUKA ODMAH POUZEĆEM, KVALITET PROVJEREN.
DUŠKO BJELOTOMIĆ, CENTAR 1, 54 550 VALPOVO, TEL. 054-82-665, 041-683-141

QL 68008

Naučite i Vi programiranje u mašinskom jeziku računara QL pomoću knjige ASSEMBLY LANGUAGE PROGRAMMING ON THE SINCLAIR QL. Ova knjiga, pisana jednostavnim i razumljivim jezikom, je bestseler za QL računare u engleskoj i nju bi trebalo da ima svaki vlasnik ovog kompjutera. Na preko 170 strana teksta nalazi se sve što je potrebno za samostalno pisanje mašinskih programa za mikroprocesor 68008. Upoznajte jednog iz najmoćnije porodice mikroprocesora! Cijena kvalitetno fotokopiranog priručnika je 2200 d. Ovdje možete nabaviti i popularni QL ADVENCED USER GUIDE po cijeni od 2600 d. Isporuka pouzećem. QL BOOK. Istarska 51, 78000 Banja Luka.

PRODAJEM JOYSTICK, palice za igre. Četiri pravca, četiri pravca ukoso i okidanje sa vrha palice. Moguć direktan priključak na „SPEKTRUM" i „galaksiju", bez interfejsa. Pošalju se detaljnim uputstvom 2100 din. POUZEĆEM. Pozivati u večernjim časovima. Stojković Dušan, Trogirski trg 2 tel. 037-29550 **37000 KRUŠEVAC**

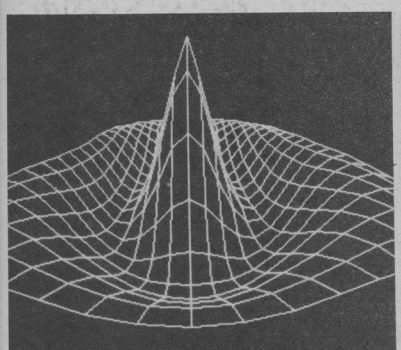
• Sve za ZX spektrum
— video i logičke igre
— namenski programi
— uputstva za programe
— knjige i preveden
— veliki izbor za Commodor BBC i QL
Garancija za , sve vrste usluga
Beta Basic 3.0, direktno iz Londona, sa kompletnim originalnim uputstvima možete dobiti samo kod COMET SOFT-a!
Milovanović Ljubiša, Petra Lekovića 57, 11030 Beograd, tel. 011/558-007.

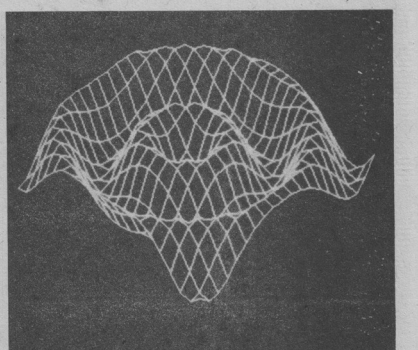
PRILIKA KOJA SE NE PROPUŠTA
BOGAT IZBOR
početnica, popularnih i stručnih knjiga na engleskom jeziku
PO NABAVNIM CENAMA
mikroprocesori
programski jezici
programiranje
operativni sistemi
spektrum
komodor 64
BBC
QL
amstrad
epl II
KNJIGE SU POTPUNO NOVE
RASPOLAŽEMO SAMO PO JEDNIM PRIMERKOM
Milovanović Ljubiša,
tel. 011/558-007

grafičko predstavljanje funkcija

Računari i matematika
ZX Spectrum

Kada je, pre tri veka, Rene Dekart uveo u matematiku pojam koordinata i koordinatnih sistema, bio je to početak nove ere u nauci uopšte. Geometrijska interpretacija procesa i funkcionalnih zavisnosti u raznim oblastima višestruko je pojednostavila često mukotrpan postupak matematičke analize. Sa jednog grafika se u trenu može uočiti ponašanje bilo koje funkcije, bez ikakvog računanja limesa i izvoda. Kućni računari raspolažu sasvim dovoljnim grafičkim mogućnostima za ovakvu primenu. Uz dobar program, jednostavno se može dobiti grafik funkcije jedne ili dve promenljive, na proizvoljnom segmentu.

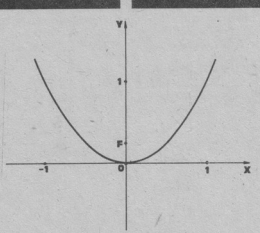




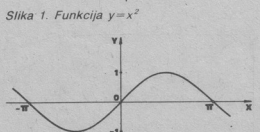
Malo matematike . . .

Funkcija je pravilo ili zakon po kome jednom skupu veličina odgovara neki drugi skup veličina. Na primer, po zakonu $y=x^2$, skup svih realnih brojeva x preslikava se u skup realnih brojeva y, tako da broju $x=2$ odgovara broj $y=4$, broju $x=3$ broj $y=9$, itd.

Sam analitički zapis $y=x^2$ nosi u sebi celokupnu informaciju o funkciji, ali ipak nije dovoljno pregledan da bi se iz njega jednostavno videla priroda veze između veličina x i y. Zbog toga se funkcije uvek predstavljaju i geometrijski, pri čemu se duž jedne koordinatne ose odmerava x, a duž druge y. Svakom x odgovara neko y, čime je određena tačka u ravni XOY. Skup svih tih tačaka daje neku krivu liniju — geometrijsku interpretaciju funkcije $y(x)$. Tako se, na primer, funkcija $y=x^2$ prikazuje *parabolom* čije je fokusno rastojanje $f=1/4$, dok, recimo, funkciji $y=\sin x$ odgovara periodična talasasta kriva sa ekstremumima $+1$ i -1, i periodom $2PI$.



Slika 1. Funkcija $y=x^2$



Slika 2. Funkcija y=sin x

. . . i malo programiranja

Grafičko predstavljanje funkcije jedne promenljive na računaru ne predstavlja gotovo nikakav problem, jer se sve praktično svodi na korišćenje naredbe **PLOT**. Ipak, o nekim stvarima se mora voditi računa.

Odmah je jasno da neće raditi jednostavan program kao što je:

```
10 FOR x=0 TO 255
20 PLOT x,SIN x
30 NEXT x
```

To što x ide od 0 do 255 je samo posledica činjenice da ekran ima 256 tačaka. Međutim, nema nikakve potrebe da i samu funkciju $y(x)$ prikazujemo na istom intervalu. Kada je u pitanju funkcija $y=\sin x$, možemo se opredeliti, recimo, za jedan period. Tada u liniju 20, umesto **SIN x**, treba staviti: **SIN (2*PI/255*x)**.

Drugi problem je što funkcija ima malu amplitudu — po modulu nije nikad veća od jedinice, pa će tako ceo grafik vrlo malo da se razlikuje od prave linije. Treba zato pomnožiti funkciju nekim faktorom, čime će se slika raširiti po ekranu. Međutim, bilo bi veoma neprirodno izvršiti množenje bilo

kojim brojem, jer će se tako izgubiti prava razmera. Ako želimo prirodan izgled grafika, moramo obezbediti da isti intervali po *x* i po *y* osi imaju istu dužinu. Pošto 255 na *x* osi vredi *2PI*, to znači da jedinica ima dužinu od *255/2PI* tačaka, i to mora biti i na *y* osi. Zato u liniji 20 treba da stoji:

255/2/PI*SIN (2*PI/255*x).

Preostaje još da se pomeri koordinatni početak i smesti, recimo, u centar ekrana. Linija 20, konačno, treba da ima oblik:

20 PLOT x,88+255/2/PI*SIN (2*PI/255*x−PI)

Bilo bi, naravno, preterano za svaku funkciju računati posebno sve ove popravke i korekcione faktore. Treba napraviti program koji će sam računati maksimum i minimum funkcije na zadatom segmentu, a onda, na osnovu toga, nacrtati uvećan grafik u prirodnoj razmeri.

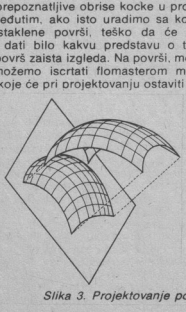
Program 1, koji mi ovde objavljujemo, jednostavan je i ima za cilj samo da pruži ideju o tome kako treba raditi sa funkcijama jedne promenljive.

Funkcije dve promenljive

Kada neka veličina *z* zavisi od dve druge nezavisne veličine *x* i *y*, onda kažemo da imamo funkciju dve promenljive. Na primer, po zakonu $z=x^2+y^2$, svakoj kombinaciji *x* i *y*, a to znači svakoj tački iz ravni *XOY*, odgovara neka vrednost *z*. Tu vrednost možemo nanositi iznad *XOY* ravni, čime dobijamo skup tačaka u prostoru. Geometrijska interpretacija funkcije *z(x,y)* je, prema tome, površ u tri dimenzije. Nije teško videti da se funkcija $z=x^2+y^2$ predstavlja *obrtnim paraboloidom* — površinom koja nastaje rotacijom parabole $y=x^2$ oko ose *y*.

Postavlja se problem kako na ekranu ili listu papira, gde raspolažemo samo sa dve dimenzije, prikazati jedan trodimenzioni objekat, kao što je površina u prostoru? Jasno, treba izvršiti nekakvo projektovanje, onako kako se to radi i kada se prikazuju geometrijska tela. Međutim, jedna površ najčešće na sebi nema nikakvih obeležja u vidu ivica ili temena, koja su inače tako karakteristična za geometrijska tela.

Zamislimo da imamo jednu kocku izrađenu od stakla. Ako tu kocku osvetlimo sa jedne strane sjalicom, a sa druge postavimo neki zaklon, senka koju budemo dobili dače prepoznatljive obrise kocke u projekciji. Međutim, ako istu uradimo sa kornadom staklene površi, teško da će nam senka dati bilo kakvu predstavu o tome kako površ zaista izgleda. Na površi, međutim, možemo iscrtati flomasterom mrežu linija, koje će pri projektovanju ostaviti svoj trag.



Slika 3. Projektovanje površi

Problem projektovanja je tako rešen. Naravno, moramo sada staklene površi, flomastere i sijalice prevesti na jezik matematike, i napraviti program za crtanje funkcija u tri dimenzije.

Mrežu linija na površi možemo dobiti u preseku sa koordinatnim ravnima *x=Const* i *y=Const*. Zatim se mogu projektovati samo čvorne tačke u mreži, dok se odsečci linija između čvorova dosta dobro zamenjuju pravim linijama. To, svakako, umanjuje kvalitet projekcije, ali višestruko ubrzava postupak projektovanja, koji bi, inače, bio neprihvatljivo spor.

Što je mreža gušća, slika je kvalitetnija, ali se tada troši daleko više vremena. Mi smo se opredelili za neku vrstu kompromisa između ova dva zahteva, tako da se mreža sastoji od ukupno 400 tačaka (20×20).

Aksonometrija i perspektiva

Postoje dva osnovna načina projektovanja. Prvi odgovara slučaju kada je izvor svetlosti, ili posmatrač, veoma udaljen, tako da su projekcioni zraci međusobno paralelni, i svi su normalni na ravan ekrana. Tačke se projektuju ortogonalno na izabranu ravan i zato kažemo da se radi o *aksonometriji*.

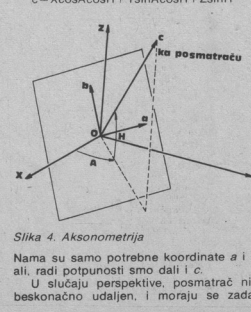
Međutim, ako je posmatrač blizu, projekcioni zraci nisu međusobno paralelni. Projektovanje je centralno, iz jedne tačke, i kažemo da se radi o *perspektivi*.

Problem koji sada moramo rešiti je sledeći:

Data je tačka *M* u prostoru, sa koordinatama *X,Y,Z*, i data je ravan *p*, na koju treba izvršiti projektovanje. U ravni *p* postoji koordinatni sistem sa osama *a* i *b*. Ako je poznat položaj posmatrača, naći formule po kojima se tačka *M* projektuje na ravan *p*, tj. po kojima mogu da se izračunaju *a* i *b*.

U slučaju aksonometrije, položaj projekcione ravni potpuno je određen ako se zada pravac ka posmatraču, a za to su dovoljna dva ugla: *azimut A* i *visina H*. Ako pravac ka posmatraču označimo sa *c*, dobijamo nov koordinatni sistem sa osama *a,b,c*. Osa *a* neka pri tome leži duž preseka ravni *p* i *XOY*. Prelazak iz jednog sistema u drugi se vrši po formulama:

$a=-XsinA+YcosA$
$b=-XcosAsinH+ZcosH$
$c=XcosAcosH+YsinAcosH+ZsinH$



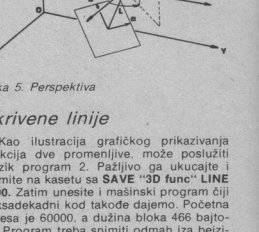
Slika 4. Aksonometrija

Nama su samo potrebne koordinate *a* i *b*, ali, radi potpunosti smo dali i *c*.

U slučaju perspektive, posmatrač nije beskonačno udaljen, i moraju se zadati njegove koordinate $X_1Y_1Z_1$, a zatim i pravac u kome on posmatra. U tački posmatranja postavićemo koordinatni sistem sa osama *a,b,c*, tako da osa *c* bude suprotno usmerena od pravca posmatranja. Uglovi *L* i *B* imaju isti smisao kao *A* i *H* u slučaju aksonometrije. Projektovanje se vrši iz tačke $X_1Y_1Z_1$ na ravan $_{p}=-1$. Izborom neke druge ravni samo bi se promenila razmera slike.

Formule za projekciju glase:

$a=-(X-X_1)\,sinL+(Y-Y_1)\,cosL$
$b=-(X-X_1)\,cosLsinB-(Y-Y_1)$
$sinLsinB+(Z-Z_1)\,cos\,B$
$c=(X-X_1)\,sosLosB+(Y-Y_1)$
$sinLcosB+(Z-Z_1)\,sinB$



Slika 5. Perspektiva

Skrivene linije

Kao ilustracija grafičkog prikazivanja funkcija dve promenljive, može poslužiti bejzik program 2. Pažljivo ga ukucajte i snimite na kasetu sa **SAVE "3D func" LINE 1000.** Zatim unesite i mašinski program čiji heksadekadni kod takođe dajemo. Početna adresa je 60000, a dužina bloka 466 bajtova. Program treba snimiti odmah iza bejzika, sa **SAVE "code" CODE 60000,466.**

Pri kasnijem učitavanju, treba samo ukucati **LOAD""**. Program se automatski startuje i crta grafik funkcije definisane u liniji 60. To je obrtna talasasta površina, nastala rotacijom krive *y=cos x* oko ose *y*.

Ukoliko u definiciji izraz za funkciju **FN z,** umesto *x i y*, stavimo *x*10.5 i y*10.5*, pomerićemo izvor talasa sa ugla slike u sam centar. A možemo eksperimentisati i sa množiteljem ispred kosinusa, čime se menja amplituda talasa.

Analiza ovog jednostavnog programa ne predstavlja naročit problem, ali čemo se mi ipak zadržati na nekim detaljima.

Projektovanje se vrši u *izometriji*, što je specijalan slučaj aksonometrije, kada se sve koordinate podjednako skraćuju. Pri tome je $A=315^{0}$, a $H=35^{0}.264$ $(sin^{-1}H=1/3)$.

Oblast na kojoj se funkcija posmatra je između 1 i 20 po *x* i *y* osi. Duž obe ose oblast je izdeljena na 19 traka, čime je formirana mreža linija 20×20, uključujući i ivične.

Treba obratiti pažnju na to kako računar iscrtava površ. Mrežu sačinjavaju četvorougaonici, kojih ukupno ima 361 (19×19). Crtaju se redom trake paralelne *x* osi, i to tako da prvo idu trake udaljenije od posmatrača. Isto tako, u okviru svake trake, prvo se crtaju udaljeniji četvorougaonici, pri čemu se još i *briše njihova unutrašnjost*. Na taj način je postignut efekat nevidljivih linija, jer bliži delovi površi zaklanjaju one dalje.

„Spektrum" nema naredbu za brisanje ili popunjavanje zatvorenih figura, pa je

```
  10 REM        PROGRAM  1
  20 REM          funkcije
  30 REM     jedne promenljive
  40 REM
  50 INPUT "Funkcija:"''"y="; LI
NE y$
  60 INPUT "interval:"''"a=";a,"
b=";b
  70 LET x=a
  80 LET max=VAL y$: LET min=max
  90 FOR n=0 TO 255
 100 PRINT AT 0,0;TAB 4-LEN STR$
 (255-n);255-n
 110 LET x=a+(b-a)/255*n
 120 LET y=VAL y$
 130 IF y>max THEN  LET max=y
 140 IF y<min THEN  LET min=y
 150 NEXT n
 160 CLS
 170 LET c=255/(b-a)
 180 IF (b-a)/(max-min)<255/175
THEN  LET c=175/(max-min)
 190 LET x=a: LET y=VAL y$: LET
xo=INT (128+(x-(b+a)/2)*c): LET
yo=INT (88+(y-(max+min)/2)*c): P
LOT xo,yo
 200 FOR x=a TO b STEP 1/c
 210 LET y=VAL y$
 220 LET x1=INT (128+(x-(b+a)/2)
*c)
 230 LET y1=INT (88+(y-(max+min)
/2)*c)
 240 DRAW x1-xo,y1-yo
 250 LET xo=x1: LET yo=y1
 260 NEXT x
```

```
  10 REM          PROGRAM  2
  20 REM            funkcije
  30 REM       dve promenljive
  40 REM
  50 DEF FN t(a,b,c,d,e,f)=USR 6
0000
  60 DEF FN z(x,y)=2*COS SQR (x*
x+y*y)
  70 DEF FN a(x,y)=(x+y-2)*6.7
  80 DEF FN b(x,y)=88+(FN z(x,y)
+(y-x)/2)*7.7
  90 DIM p(20,2): DIM q(20,2)
 100 FOR x=1 TO 20
 110 LET p(x,1)=INT (FN a(x,20)+
0.5)
 120 LET q(x,1)=INT (FN b(x,20)+
0.5)
 130 NEXT x
 140 FOR y=19 TO 1 STEP -1
 150 FOR x=1 TO 20
 160 LET p(x,2)=p(x,1)
 170 LET q(x,2)=q(x,1)
 180 LET p(x,1)=INT (FN a(x,y)+0
.5)
 190 LET q(x,1)=INT (FN b(x,y)+0
.5)
 200 NEXT x
 210 FOR x=2 TO 20
 220 RANDOMIZE FN t(p(x,2),q(x,2
),p(x-1,2),q(x-1,2),p(x-1,1),q(x
-1,1))
 230 RANDOMIZE FN t(p(x,2),q(x,2
),p(x,1),q(x,1),p(x-1,1),q(x-1,1
))
 240 PLOT p(x,2),q(x,2)
 250 DRAW p(x-1,2)-p(x,2),q(x-1,
2)-q(x,2)
 260 DRAW p(x-1,1)-p(x-1,2),q(x-
1,1)-q(x-1,2)
 270 PLOT p(x,2),q(x,2)
 280 DRAW p(x,1)-p(x,2),q(x,1)-q
(x,2)
 290 DRAW p(x-1,1)-p(x,1),q(x-1,
1)-q(x,1)
 300 NEXT x
 310 NEXT y
 320 STOP
1000 CLEAR 59999: LOAD ""CODE 60
000: BORDER 1: PAPER 1: INK 7: R
UN
```

```
OTKUCAJTE PRVO HEX LOADER, A ZA-
TIM POMOCU NJEGA HEX LIST. KADA
KUCATE HEX LIST, ENTER PRITISNI-
TE TEK NA KRAJU SVAKE LINIJE.
PRVIH OSAM BAJTA IZ SVAKE LINIJE
IDU U MEMORIJU, A DEVETI SLUZI
KAO CHECKSUM.
```

```
  10REM CTRL CODE HEX LOADER

  20 BORDER 1: PAPER 1: INK 7: P
OKE 23609,20
  30 CLEAR 26999: POKE 23658,8:
PRINT AT 20,0
  40 INPUT "START ADDR:";A
  50INPUT (A);": ";LINE A$:IF A$
=""THEN STOP
  60 LET C=0: IF LEN A$<>18 THEN
  GO TO 90
  70 FOR I=1 TO 17 STEP 2: LET X
=16*CODE A$(I)-768-112*(A$(I)>"9
")+CODE A$(I+1)-48-7*(A$(I+1)>"9
"): IF I<17 THEN  POKE A,X: LET
A=A+1: LET C=C+X
  80 NEXT I: LET A=A-8: IF C-256
*INT (C/256)=X THEN  PRINT A;":
";A$: LET A=A+8: GO TO 50
  90 PRINT #1; FLASH 1;"ERROR!":
 BEEP 1,1: GO TO 50
```

```
MC
CTRL CODE HEX LIST

60000: 219BEAE52A0B5CDDF9
60008: 211FEC3E062323CD83
60016: B433233D20F7CD0732
60024: 23C5CD0723C5CD0778
60032: 23D1E13EAF9047DA73
60040: F9243EAF9257DAF9C6
60048: 243EAF9467DAF92403
60056: C34EEB215827D9C93E
60064: 78A71F371FA71FA802
60072: E6F8A867790707077B
60080: A8E6C7A807076F79F3
60088: E607C90C833803BA3A
60096: 38F99205C905833851
60104: 02BAD8920CC9058383
60112: 3802BAD8920DC90D41
60120: 833803BA38F9920540
60128: C90D833803BA38F97F
60136: 9204C904833802BADA
60144: D8920DC90483380201
60152: BAD8920CC90C8338C0
60160: 03BA38F99204C9E532
60168: 78BC3E0F38012F5740
60176: 79BD3E3C38012FA2BA
60184: 577C903002ED44672D
60192: 3C321FEC7D913002B9
60200: ED446FBC3E6638043C
60208: 2F5D6C63A216FF1426
60216: 0F30FC7A875F1600B1
60224: E52122EC195E235604
60232: EBD17A1FC1C9AFED7B
60240: 42C809AFED52C819E2
60248: EBAFED42C8097CBAD0
60256: 3801EB7CB8380560F5
60264: 477D694F7AB83805EB
60272: 5047794B5F7CBA2818
60280: 0978BA2071446779F0
60288: 4D6F7CB8C87DBB3828
60296: 01EBD5C5CD07EB2267
60304: A8EB08D9C1E1CD07EA
60312: EB22A3EBD908180A9E
60320: 08D9CDBBEAD908CD01
60328: BBEAF5CDA0EAC5D58B
60336: A72808473EFFCB3F65
60344: 10FC2F4708D9F5CD25
60352: A0EAC547043EFFCBA2
60360: 3F10FCE5D94FD17DA6
60368: 93782809A6772C7D02
60376: 933E0020F7B1A677B6
60384: D9C1D9F108D1C1F1EF
60392: DD3500C818B22220E6
60400: ECC5D5CD07EB22FB62
60408: EBE1CDBBEA321FEC7B
60416: 7CB83A1FEC20F359E5
60424: 50C1ED52C81938016A
60432: EBE5D5CD8AEBD1E199
60440: ED4B20ECC38AEB007C
60448: 0000BBEAC5EACEEA0C
60456: D7EAE1EAEBEAF4EA3F
60464: FDEA000000000000E7
```

zato bejziku dodat mašinski program na adresi 60000. Pozivanje se vrši pomoću funkcije **FN p**, sa šest parametara: *x* i *y* koordinatama tri tačke na ekranu. Program obavlja brisanje unutrašnjosti trougla sa zadatim temenima. Pošto mi imamo četvorougaonike koje treba izbrisati, podelićemo ih dijagonalama na po dva trougla, i dva puta pozvati program za brisanje trouglova.

Nakon brisanja vrši se iscrtavanje stranica četvorougaonika. Većinu tih linija računar povlači dva puta, jer se radi o zajedničkim stranicama za dva susedna četvorougla. Algoritam je rešen tako da u oba prolaza iscrtavanje ide u istom smeru, što je urađeno čisto iz estetskih razloga, jer bi inače neke linije bile neprirodno zadebljane (linije koje „spektrum" crta u dva smera ne poklapaju se uvek, iako su polazna i krajnja tačka iste).

Mašinac protiv bejzika

Program 2 nas svakako ne može zadovoljiti svojim mogućnostima. On iscrtava uvek istu funkciju, na istoj oblasti promenljivih, posmatrano iz istog pravca — i za to mu je potrebno čak puna tri minuta vremena. Kako bi tek bilo kada bismo napravili program sličan programu 1: da po volji unosimo analitički oblik funkcije, oblast na kojoj se vrši projektovanje, vrstu projektovanja i položaj posmatrača, a računar sam da izračuna sve ostalo i iscrta grafik u prirodnoj razmeri?

To se, svakako, može izvesti i u bejziku, ali nam ne zvuči privlačno. Zato smo, specijalno za ovu priliku, razradili jedan ozbiljan mašinski program za projektovanje funkcija u tri dimenzije. Program, zajedno sa uputstvima, možete naručiti od redakcije „Galaksije", a ovde ćemo samo pomenuti neke od njegovih značajnijih karakteristika.

Smatrali smo da je vreme izvršavanja primarni zahtev, tako da na štednju memorije uopšte nije ni obraćana pažnja. Program zauzima 14 kilobajta, ali više od polovine otpada na razne tabele i sadržaje ekrana. Takođe, veliki deo zauzima kalkulator za mantisu od 24 bita, umesto 32 bita, koliko obrađuje „spektrumov" originalni kalkulator. Tačnost je tako neznatno smanjena, ali je izvršavanje bitno ubrzano. Svega dvadesetak sekundi potrebno je za iscrtavanje iste one kosinusne funkcije, koju smo već imali u programu 2, pri čemu se još može jednostavno menjati ugao iz koga se posmatra, oblast promenljivih, tip projektovanja (aksonometrija ili perspektiva), itd.

Program je opremljen jednostavnim ekranskim editorom. Svi parametri, ne vodeći računa o njihovom redosledu, ukucavaju se u tabelu, koja se nalazi na ekranu. Zatim će, na pritisak samo jednog tastera, računar pročitati sadržaj tabele i prikupiti sve neophodne informacije: analitički oblik funkcije, oblast u ravni *XOY* i položaj posmatrača. Malo ćemo sačekati dok se računaju projekcije svih 400 tačaka u mreži, a onda će se na ekranu iscrtati zadata funkcija u zahtevanom položaju.

Pojedini crteži mogu biti zaista efektni. Preporučujemo vam zato da nabavite ovaj program čak i ako vas matematika naročito ne interesuje.

Jovan Skuljan

Nagradni zadatak

Kada izlaze „Računari 101“?

Nagradni zadaci, nekada omiljena zabava „Galaksijinih“ čitalaca, nisu bili baš čest gost u „Računarima“. Verujući da će se programeri rado takmičiti u nečemu što je jednostavnije od pisanja priloga za naše konkurse, odlučili smo da iz broja u broj objavljujemo probleme koje ocenjujemo kao umereno jednostavne.

ZADATAK 1

*Časopis „Računari“ izlazi svakog drugog meseca (10. februar, 10. april, 10. jun . . .). Sastavite program koji će, za zadato N, izračunavati u koji **dan** (npr. ponedeljak, sreda . . .) izlaze „Računari N“. Uz program napišite i u koji će dan izaći „Računari 101“.*

***Među** tačnim rešenjima ovoga zadatka **koja su**, zajedno sa kuponom, pristigla u **Redakciju** pre 1. novembra 1985, žrebom će biti izvučene tri novčane **nagrade**: prva nagrada od 10.000 dinara, druga od 5000 i treća od 3000 dinara. Prva nagrada može da bude dodeljena i bez udela boginje Fortune ukoliko se neki program bude izdvajao kvalitetom.*

Rezultati će biti objavljeni u „Računarima 9“

Dejan Ristanović

Rešenje nagradnog zadatka 1

„Računari 101“ izlaze u _____

Ime i prezime _____

Godina rođenja _____

Adresa _____

Mesto _____

Hajde da se varamo

Pokice za život

U tabeli su dati POKE-ovi za neke najnovije igre za „spektrum“, ali je za njihovo unošenje potrebno bar minimalno poznavanje mašinskog jezika. Pošto je većina novijih programa, koji kruže YU-soft scenom, snimljena „headerless“ (bez zaglavlja) i sličnim sistemima zaštite, „pokovanje“ iz bejzika je ponekad nemoguće zbog dužine bloka koji se učitava.

Izgleda da je našim „rasturačima“ programa (ovaj izraz smatram adekvatnijim od rasprostranjenog izraza „pirati“) mnogo veća briga kako da zaštite svoja imena i telefonske brojeve, koje tako nametljivo ubacuju u programe (inače, te zaštite su veoma primitivne), nego da omoguće nesmetano ubacivanje POKE-ova za bezbroj života u igre. Tu bi „rasturači“ programa za „spektrum“ mogli da se ugledaju malo na „komodorovce“.

Nije redak slučaj ni da rasturen program ne radi kako treba — ovde bi mogli navesti i imena krivaca — ali, na sreću, oni sami su se pobrinuli da ne ostanu anonimni.

U svemu ovome, ipak, ima i nečeg dobrog — mučenja oko ubacivanja, ili traženja „pokica“ za olakšavanje igre je dobra prilika da se upustite u svet mašinca.

Većina POKE-ova je data u NEX-kodu, zbog toga što je za njihovo unošenje potreban nekakav monitor ili disasembler. Tamo gde je POKE dat decimalno, vrlo je verovatno, da je moguće unošenje iz bejzika na već mnogo puta opisan način. (Zavisi od verzije koju ste dobili).

Za one koji slabo poznaju mašinski jezik, evo nekoliko saveta: učitavanje blokova bez zaglavlja je ostvareno najčešće na sledeći način:

LD IX, početna adresa bloka
LD DE, dužina bloka
LD A, fleg, — najčešće FF
SCF; signal za LOAD, a ne VERIFY
CALL &556; rutina za učitavanje
JP Start; skok u program

POKE-ove treba ubaciti između CALL &556 i JP Start, najjednostavnije je naredbama:

LD A, nov sadržaj
LD (ADRESS), A

gde je ADRESS adresa data u tabeli za određenu igru. Ukoliko se učitava više blokova, upis treba da dođe iza zadnjeg bloka. Ovde nije na odmet i ubaciti kontrolu ispravnosti učitavanja, koja nedostaje kod mnogih programa. Ona treba da dođe nakon CALL &556 i to:

JR C, CONT
RST 8
DEFB 26; Tape loading error
CONT; Nastavak upisa ili start

Kod nekih igara je stvar malo komplikovanija, pošto je loader sakriven, ili zbog manipulacija sa stekom. No, nemoguće je dati savete za sve varijante zbog ograničenog prostora, kao i zbog sve novijih metoda zaštita. Ako u tabeli nije data primedba, znači da je POKE za bezbroj života.

Petar Putnik

TABELA

Beaky &the Egg Snatchers	31948,201 (Magic/time)
Black Hawk	34695,183
Chuckie Egg 2	65535,176 (POKE može i pre LOAD)
Robotron	54158,0
Spy Hunter	54824,0
Wizard's Lair	47968,0
Astronut	&BC54,0
Bear George	&D8F5,0
Boulder Dash	&7920,&B6
Brian Bloodaxe	&67D6,0
Bruce Lee	&CA53,0
Dark Star	&D175,&C9
Death Star Interceptor	&95CC,0
Falcon Patrol II	&9E69,&A6
Gilliagan's Gold	&CEA2,&10
Great Escape	&DC99,&B7
HERO	&ADE9,&B6—Životi &D686,0—Bombe
JetSet Willy II	&6F12,&4F &6F13,&60 &6F17,&99 bezbroj života &6EEF,&9B potpuna neranjivost
Ms Pacman	&CE97,0
Number 1	&E109,0
Psytron	&66B8,&C9 &9C9C,0 bezbroj disruptera
Sir Lancelot	&5D55,&A6
Starion	&B609,&B6
Tapper	&81D1,&B6
Witch Cauldron	&9C78,0
Worse Things Happen	&8A73,0

Nove knjige

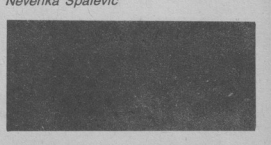
„Commodore I/0“

Konačno se u izlozima naših knjižara pojavila prva knjiga na srpskohrvatskom jeziku posvećena „komodor“ računarima. To je „Commodore I/0“ autora mr Lidije i Momira Popovića u izdanju Beograd biroa. U knjizi su data uputstva za rad i programi koji olakšavaju korišćenje kasetofona, disk jedinice i štampača. Najveću vrednost knjizi daju poglavlja posvećena radu sa sekvencijalnim, random (rasutim) i relativnim datotekama koja se u sistematizovanom obliku ne mogu naći na jednom mestu ni u stranoj literaturi. Uz to nisu izostavljene ni baze podataka, a dat je i prikaz najboljeg programa ovog tipa za „komodor 64“ — Superbase.

Mada se autorima moraju uputiti i poneke zamerke na korišćenu terminologiju i stil izlaganja, možemo preporučiti ovu knjigu našim čitaocima i poželeti da se što pre i što više knjiga poput ove, posvećenih računarima i njihovoj primeni na našem jeziku, nađe u knjižarama.

Knjiga „Commodore I/0“ košta 1500 dinara. Ako je nema u vašim knjižarama, možete je poručiti preko jednog od telefona 011-667-703, 011-134-180, 011-821-856, ili naručdžbenicom na adresu: Beograd biro, Trg Lole Ribara 32a, 11400 Mladenovac.

Nevenka Spalević



Računari iz mog ugla

kako preživeti računare

Ako mislite da se računari mogu koristiti samo za igranje, programiranje i poslovnu primenu, ljuto se varate. Postoji sedam stadijuma računarskog ludila, kao kod Šekspira sedam životnih doba, i svaki vam može pružiti izvesne radosti i koristi.

Ti stadijumi su: 1) infantilni, 2) igrački, 3) hakerski, 4) programerski, 5) pisački, 6) naučni i 7) apstraktni. U svakom od njih mogu se naći ljudi različitih životnih dobi, no logično bi bilo da postoji gradacija.

1) Na prvom nivou nalaze se protivnici računara. Ograničićemo se na proučavanje reprezentativnog predstavnika ove vrste: Negatio Tipicus Vulgaris. On se inati kao razmaženo dete koje odbija da jede bilo šta drugo osim slatkiša. On je u zamišljenom sukobu sa računarima jer im nije dorastao. Godi mu razmišljanje o pobedi u neravnopravnoj borbi sa nadmoćnijim neprijateljem. On mašta o epskom sukobu golog mišića i računara. Mi smo se uvek razmetali kako u neravnopravnoj borbi pobeđujemo i desetostruko jačeg neprijatelja. Ostavimo to štivo za čitanke.

Međutim, on ipak realno posmatra stvari. Njemu je najveći užitak njegov posed, dokazivanje socijalnog statusa. Treba po svaku cenu posedovati vikendicu, automobil, par metara knjiga, ako treba i računar. Nabaviše ga da posle neko ne bi mogao da kaže da mu loše ide. Neku korist od računara može izvući jedino ako ih šverc uje.

2) Samo jedan mali korak potreban je za prelazak u sledeći nivo. Treba samo da pokušate i postaćete neumorni utamanjivač napadača iz svemira. Za klince je prirodno da se nalaze u kategoriji igrača, a šta sa ozbiljnim ljudima? Oni ne mogu tek tako da podrivaju svoj autoritet, i zato, dok se njihovi sinovi igraju danju, oni noću ganjaju sprajtove po ekranu. Poznaćete ih ujutro po zakrvavljenim očima. Na ovom stadijumu kompjuterska boleština postaje zarazna i veoma se lako prenosi. Izvesna korist izvlači se iz razmene i preprodaje programa, tek za zadovoljenje narkomanskih strasti.

3) Normalna stvar je da posle izvesnog igračkog staža počinjete da čačkate po programima, da ih menjate, prilagođavate i razbijate zaštitu. To je hakerski stadijum; nastavljate da se igrate, ali po sopstvenim pravilima. Uskoro ćete početi da prčkate i po hardveru. Pred vama je izazov — treba da pokvarite sopstveni računar.

Hakeri su pravi zaluđenici, večito na šteti, ako se ne uzme u ozbir njihovo unutrašnje zadovoljstvo. Važan deo hakerskog zanata je sposobnost da dobro barata žargonom, i da ume dovoljno slikovito i napaljeno da objasni svoj podvig.

Hakeri obično imaju megalomanske planove koji u priči lepo zvuče. Čim izgubi spontano zezalačko zadovoljstvo prelazi se u neki drugi stadijum gde korist ide ispred zadovoljstva. Zapravo, uživa se u koristi. Tu se može povući granica između amaterizma u lepom smislu i profesionalizma u ružnom. Od lepog hakera postaje ružan profesionalac. Kao ono sa leptirima i gusenicama, samo obrnuto.

4) Iz neozbiljnog odjednom se prelazi u hiperozbiljno stanje. Kao dečaci kad odjednom odbijaju da nose kratke pantalone. Programeri su nosioci teze da je računar ozbiljna naprava, nikako igrarija. Bave se programiranjem u vidu zanata; po mogućstvu mašinac, ili bar tako da izgleda.

Na hardverskom polju oni primenjuju olimpijsko geslo: brže, više, jače. Glavna preokupacija im je nabavka što moćnijih mašina. To je izvor njihove nesreće, ali i goleme sreće, kada uspeju za trenutak da imaju šta drugi nemaju. Baš kao i bilderi, imaju neodoljivu potrebu za šepurenjem i prikazivanjem: mišići moraju da budu što veći, pa makar oni i nemali dovoljno krvi da ih napune.

5) Pisci dolaze iz prethodne grupe, mada ima i padobranaca koji dolaze niotkuda. Na primer, ne znaju ništa o računarima, ali su pismeni i udružuju svoj rad sa onima koji ništo znaju, ali ne umeju da pišu. Oni su hiperproduktivni; ne opterećuju se stvarima o kojima pišu, nego razmišljanjima o lovi.

Ovde spadaju i prosvetitelji, koji udeluju narodu onih par bejzik instrukcija što su pogrešno naučili za kratko vreme dok su boravili u prethodnom stadijumu. Bejzikodrži drže se ko braća. Jedni drugima pišu recenzije i prikaze objavljenih knjiga. Hipersuperultra brzi kursevi bejzika nisu otimačina, nego plemeniti poziv. Eto prilike da i Fantom osavremeni svoj metod pa da u liftu između prizemlja i osmog sprata usamljenim ženskim osobama drži kurs bejzika. Očita bi bila korist od takvih koji bi na brz i efektan način računarski opismenili naše domaćice.

Pisačka strast se naročito neguje kod frustriranih koji nemaju formalnih uslova da napreduju u hijerarhiji.

6) Da biste učestvovali u igri na šestom nivou, morate ispunjavati određene uslove čisto formalne prirode. Ali, zato je bavljenje kompjuterskom naukom naročito lep imidž: uliva strahopoštovanje običnom čoveku koji ne može da prati šta se sve tamo gore dešava.

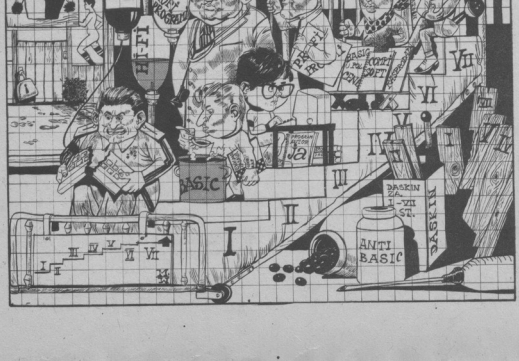
Nema sumnje, bavljenje naukom je mukotrpan posao, koji osim socijalnog prestiža ne donosi baš mnogo koristi. Zato će penjači od karijere nastojati da protutnje kroz ovaj stadijum kao kroz tranzitnu stanicu i ulete u sigurne vode sedmog, poslednjeg stadijuma.

7) Kod transcedentalaca na apstraktnom nivou, baš kao i kod klinaca, postoji velika sklonost ka priči. Impotencija, teoretisanje, drugo detinjstvo. Biti što zamršeniji i nerazumljiviji. Oni koji se bave računarima na nebu propustili su priliku da prodru kroz prethodne stadijume, a naročito igranje, već su se preko reda implantirali na transcedentalni nivo. Oni ne mogu imati širinu posmatranja problema malih računara ako im fali tako veliko parče kao što je igrarija.

Oni koji se maksimalno užive u ulogu naučnika dolaze u ovu poslednju fazu. Ta divna stvorenja više ne znaju šta se oko njih dešava, ali im to nije ni potrebno. Transcedentalna apstrakcija je kidanje veza sa realnošću, uvod u novi infantilni period. Krug se zatvara.

Naravoučenije: frustrirana deca koja nisu imala prilike da se igraju mogu postati veliki namćori.

Jelena Rupnik



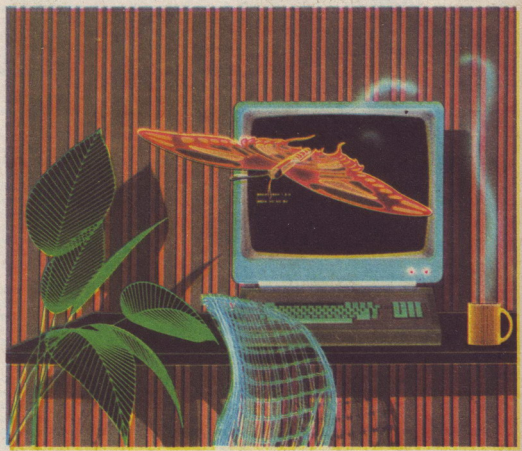
na vrh brda vrba mrda

Animacija ekrana

Prozori na kockice

Prozori (windows) koji se mogu dobivati pri ekranskom prikazu značajni su ne samo za poslovnu grafiku već se mogu koristiti i u igrama — na primjer za prikaz bacanja kocaka. Sljedeći program ilustrira primjenu prozora u igrama:

```
  5 REM bacanje kocaka
 15 MODE 0
 25 INK 0,2
 35 BORDER 18
 45 WINDOW #1,3,9,5,18
 55 WINDOW #2,12,18,5,18
 65 INK 2,25 : INK 3,15
 75 INK 4,8 : INK 5,6
 85 PAPER#1,2 : PAPER#2,4
 95 PEN#1,3 : PEN#2,5
115 CLS#1 : CLS#2
125 RANDOMIZE TIME
135 a=INT (RND*6+1)
145 b=INT(RND*6+1)
155 SOUND 1,1,27,0,0,2
165 IF a=1 THEN GOSUB 375
175 IF a=2 THEN GOSUB 415
185 IF a=3 THEN GOSUB 375 : GOSUB
415
195 IF a=4 THEN GOSUB 415 : GOSUB
465
215 IF a=5 THEN GOSUB 375 : GOSUB
415 : GOSUB 465
225 IF a=6 THEN GOSUB 415 : GOSUB
465 : GOSUB 525
235 IF b=1 THEN GOSUB 575
245 IF b=2 THEN GOSUB 615
255 IF b=3 THEN GOSUB 575 : GOSUB
615
265 IF b=4 THEN GOSUB 615 : GOSUB
665
275 IF b=5 THEN GOSUB 575 : GOSUB
615 : GOSUB 665
285 IF b=6 THEN GOSUB 615 : GOSUB
665 : GOSUB 725
295 c=INT(RND*3*)
315 IF c=1 THEN 345
325 CLS#1 : CLS#2
335 GOTO 135
345 d345 d$=INKEY$ : IF d$<> CHR$(32)
THEN GOTO 345
355 CLS#1 : CLS#2
365 GOTO 135
375 LOCATE#1,4,7 : PRINT#CHR$ (143)
385 LOCATE#1,4,8; print#1,CHR$ (143)
395 RETURN
415 LOCATE#1,6,3 : PRINT#1,CHR$(143
425 LOCATE#1,6,4 : PRINT#1,CHR$(143)
435 LOCATE#1,2,11 :
PRINT#1,CHR$(143)
445 LOCATE#1,2,12 :
PRINT#1,CHR$(143)
455 RETURN
465 LOCATE#1,2,3 : PRINT#1,CHR$(143)
```



```
475 LOCATE#1,2,4 : PRINT#1,CHR$(143)
485 LOCATE#1,6,11 :
PRINT#1,CHR$(143)
495 LOCATE#1,6,12 :
PRINT=1,CHR$(143)
515 RETURN
525 LOCATE#1,2,7 : PRINT#1,CHR$(143)
535 LOCATE#1,2,8 : PRINT#1,CHR$(143)
545 LOCATE#1,6,7 : PRINT#1,CHR$(143)
555 LOCATE#1,6,8 : PRINT#1,CHR$(143)
565 RETURN
575 LOCATE#2,4,7 : PRINT#2,CHR$(143)
585 LOCATE#2,4,8 : PRINT#2,CHR$(143)
595 RETURN
615 LOCATE#2,6,3 : PRINT#2,CHR$(143)
625 LOCATE#2,6,4 : PRINT#2,CHR$(143)
635 LOCATE#2,2,11 :
PRINT#2,CHR$(143)
65 LOCATE=2,2,12 : PRINT#2,CHR$(143)
655 RETURN
665 LOCATE#2,2,3 : PRINT#2,CHR$(143)
675 LOCATE#2,2,4 : PRINT#2,CHR$(143)
685 LOCATE#2,6,11 :
PRINT#2,CHR$(143)
695 LOCATE#2,6,12 :
PRINT#2,CHR$(143)
715 RETURN
725 LOCATE#2,2,7 : PRINT#2,CHR$(143)
735 LOCATE#2,2,8 : PRINT#2,CHR$(143)
745 LOCATE#2,6,7 : PRINT#2,CHR$(143)
755OCATE#2,6,8 : PRINT#2,CHR$(143)
765 RETURN
```

Dva prozora koja se koriste u programu predstavljaju dvije kocke. Mod 0 je odabran iz razloga što visoka grafička rezolucija nije potrebna. Linija 25 postavlja boju „papera" na svijetlo plavo, a linija 35 daje zelenu pozadinu. Nakon tih linija definirani su prozori. Na slici 1. su prikazane pozicije prozora i točaka na kockama. Kako se može vidjeti, svaki je prozor veličine 7×14, što konačno daje kvadratičan oblik kocaka. Točke na kockama su raspoređene nizom instrukcija IF . . THEN te potprogramima. Po četiri potprograma se koriste za svaku kocku. Kod primjene prozora važno je znati da se ne koriste uobičajene „screen" koordinate, već da prozori imaju vlastite koordinate.

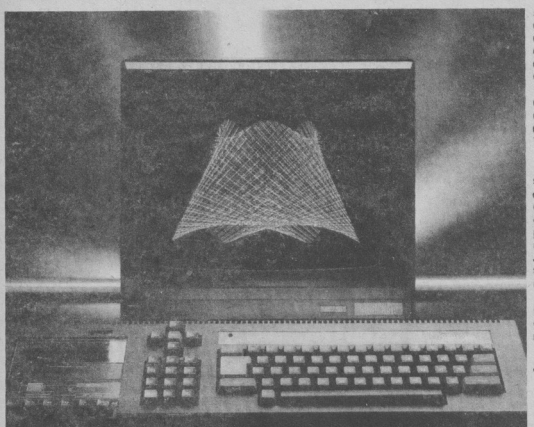
Igra pomičnih znakova

Kroz igre se najlakše upoznje s računalima pa tako i s njihovom grafikom. Može se jednostavna igra, kao na primjer, hvatanje određenih predmeta (znakova) na ekranu. U ovoj igri koja ilustrira grafiku pomičnih znakova treba uloviti znak — srce — za što će se dobiti 10 poena, a ako se uhvati strelica određen broj poena se oduzima. Primjer igre:

```
 15 MODE 1
 25 BORDER 18 : PAPER 2 : CLS
 35
```

Programiranje u bejziku AMSTRAD/SCHNEIDER

U prošlom broju „Računara" opisani su neki osnovni načini upotrebe grafičkih mogućnosti računala Amstrad/Schneider. Potpuno savladavanje i iskorištenje grafičkih potencijala ovog računala nije moguće bez naprednijeg programiranja u npr. strojnom kodu (machine code). Međutim, da bi se stiglo do te vrste programiranja, potrebno je savladati cjelovitije programe u lakše shvatljivom i preglednijem bejziku. Analizirajući programske linije od kojih su programi sastavljeni, može se otkriti i naučiti princip animacije grafike — pomicanje, prolaženje jednog objekta ispred ili iza drugog i rotiranje



```
a$=CHR$(32)+CHR$(205+"_"
+CHR$(2Ø4)+CHR$(32
 45 b=17 : c=24
 55 LOCATE 1,1 : PRINT Ø : PRINT
 REZULTAT"
 65 LOCATE 34,1 : PRINT "NAJREZ"
 75 RANDOMIZE TIME
 85 GOSUB 625
 95 FOR d=1 TO 2Ø
115 IF RND> Ø.5 THEN e$=CHR$(228) :
f=3 : ELSE e$=CHR$(241) : f=1
125 g=INT(RND*25)+7
135 GOSUB 245
145 IF(g=b+2) OR (g=b+3) THEN
GOSUB 395
155 LOCATE h,i : PRINT CHR$(32)
165 NEXT d
175 GOSUB 455
185 GOSUB 555
195 IF LOWER$(j$)="d" THEN 85
215 CALL &BBFF
225 PAPER Ø : PEN 1 : CLS
235 END
245 REM rutina za igru
255 h=7 ; i=1
265 FOR k=1 TO 23
275 LOCATE h,i : PRINT CHR$(32)
285 LOCATE g,k :PEN f,: PRINT e$
295 h=g : i=k
315 SOUND 1,k* 5Ø,5
325 IF NOT INKEY(71) THEN b=b-1
335 IF NOT INKEX(22) THEN b=b+1
345 IF b<1 THEN b=1
355 IF b>34 THEN b=34
365 LOCATE b,c : PEN Ø : PRINT a$
375 NEXT k
385 RETURN
395 REM bodovanje
415 IF f=3 THEN m=m+1Ø : SOUND
1,5Ø,5Ø
425 IF f=1 THEN m=CINT (m/2) : SOUND
1,1ØØØ,1ØØ
435 LOCATE 1,2 : PEN Ø : PRINTm
445 RETURN
455 LOCATE 15,1Ø
465 PEN 3
475 WHILE INKEY$<>"" : WEND
485 PRINT „Nova igra ? (d/n)"495 .
j$=INKEY$ : IF j$="" THEN 495
515 LOCATE 15,1Ø
525 PRINT SPACE$(18)
535 if lower$(j$="Z" OR j$="'"THEN
455
545 RETURN
555 IF m>n THEN n=m
565 LOCATE 34,2
575 PEN Ø
585 PRINT n
595 m=Ø
615 RETURN
625 REM startna rutina
635 LOCATE 1Ø,1Ø
645 PEN 3
655 WHILE INKEY$,,"" : WEND
655 WHILE INKEY$<>"" : WEND
675 p$=INKEY$ : IF p$="" THEN 675
685 LOCATE 1Ø,1Ø
695 PRINT SPACE$(22)
7Ø5 RETURN
```

53/na vrh brda vrba mrda

U programu linije 15 do 65 definiraju string hvataljke te prikaz rezultata. Linija 115 odlučuje koji će od 2 znaka biti prikazan (srce ili strelica). Kad neki od ta dva znaka dođe do dna ekrana linija 145 uspoređuje položaj znaka i hvataljke. ako je znak uhvaćen, poziva se rutina sa lokacije 395. Jedna serija obuhvaća 20 znakova, a poslije toga rutina sa linije 455 otvara mogućnost nove ire.

Animacija u kojoj se koriste tekst koordinate je vrlo jednostavna, a preoručljivo je da čitaoci mijenjaju tu igru prema svom nahođenju, učeći stvaranje igara.

Slova po ukusu

Amstrad/Scneide može prikazati 256 različitih znakova. Prvih 32 nisu „vidljivi" već su dani kao kontrolni kodovi. Mogu se vivideti ako se upotrebi kratka rutina:

```
15 FOR A=Ø TO 31
25 PRINT CHR$(1);
35 PRINT CHR$(A);
45 NEXT A
55 FOR A=32 TO 255
65 PRINT CHR$(A)
75 NEXT A
85 END
```

Postojeći set od 256 znakova se može zamijeniti potpuno drugim setom. Pri tome se koriste instrukcije SYMBOL i AFTER. SYMBOL AFTER 0 znači da se želi redefinirati svih 256 znakova-karaktera. Svaki znak je definiran kao rutina od 8×8 točaka, a postavlja se sa 1 za ON i 0 za OFF. Na primjer slovo A—CHR$(65) može – ti definirano kao heksadecimalni niz vrijednosti:

SYMBOL 65, &18, &3C, &66, &7E, &66, &66,Ø

što bi prikazano grafički imalo izgled:

```
. . * * . . . .   &18
. . * * * * . .   &3C
. * * . . * * .   &66
. * * . . * * .   &66
. * * * * * * .   &7E
. * * . . * * .   &66
. * * . . * * .   &66
. . . . . . . .   &00
```

Treba obratiti pažnju na činjenicu da točka (pixell) stvarno odgovara točki u modu 2. U

modu 1 točka je u stvari 2 točke, a u modu 0 točkā je 4 točke.

U sljedećem primjeru programa koristi se redefinirani znak oblikovan prema želji korisnika, koji ima oblik tenka. (Slika 2.) Program je izveden u modu 0, a koristi se karakter CHR$(254). Tenk će se kretati prikazanim pejzažem, a takva grafika i način kretanja može se naći u mnogim igrama. Za definiranje redefiniranog karaktera može se služiti binarnim, heksadecimalnim ali i dekadskim brojevima kao u ovom primjeru (linija 75). Ukoliko se radi s binarnim ili heksadecimalnim brojevima, za njihov upis se koristi oblik &X.

Primjer programa:

```
 15 MODE 0
 25 INK 3,12 : INK 9,0
 27 INK 11,0
 35 INK 1,3 : INK 5,3
 37 INK 13,3
 45 PAPER 3 : CLS
 55 WINDOW#1,1,20,1,10
 65 PAPER#1,6 : CLS#1
 75 SYMBOL
254,0,0,31,24,126,255,255,126
 85 a=5: b=25@ : c=15@ : GOSUB 175
 95 a=1 : b=45@ : c=2@@ : GOSUB 175
115 PRINT CHR$(23);CHR$(1);
125 GOSUB 265
135 PRINT CHR$(23);CHR$(0);
145 WHILE INKEY$="" : WEND
155 MODE 1 : CALL &BBFF : PAPER 0
165 END
175 REM crtanje piramida
185 d=2@@
195 FOR e=1 TO 1@@
215 MOVE b-d/2,c+e
225 DRAWR d,0,a
235 d=d-2
245 NEXT e
255 RETURN
265 TAG
275 PLOT# 0,0,8
285 c=212 : f=-64
295 FOR b=0 TO 64@ STEP 4
315 MOVE f,c
325 CALL & BD19
335 PRINT CHR$(254);
345 MOVE b,c
355 CALL &BD19
365 PRINT CHR$(254);
375 f,b
385 FOR g=1 TO 1@@ : NEXT
395 NEXT b
415 TAGOFF
425 RETURN
```

U ovom programu se može primjetiti da je moguće postaviti tekst karakter na mjesto tekst kursora. Tada se karakter može precizno pomicati točku po točku, dajući tako fino kretanje. Instrukcije za to su TAG i TAGOFF. Pri upotrebi instrukcije TAG boja znaka nije upravljana s PEN. Program izvodi sliku: plavo nebo, žuti pijesak te crvene piramide. Mali crni tenk pomiče kroz pejzaž iza prve piramide i ispred druge piramide. Za ostvarenje toga potrebno je resetirati broj boja „penova". To se izvede tako da se odabere broj od 8 ili više penova za karaktere, a broj manji od 8 za pozadinu. U ovom programu pen 8 je korišten za tenk. Za piramide odabrani su različiti penovi : 5 za prvu i 1 za drugu (linja 35). Prolaženje tenka ostvareno je linijama 25 i 37, a brzina je definirana u liniji 385.

U prethodnom primjeru moglo se učiti postojanje „dubinskog" prikaza u više planova: pozadina, sredina i prednji plan. Inače, uobičajeni prikaz ima 4 različita plana: pozadina, objekt u sredini koji prekriva pozadinu, objekt u prednjem planu koji prekriva pozadinu te prednji plan koji prekriva sve ostalo. U modu 1 imamo 4 različita „inka". Može se zapitati kako koristiti ink 0 za pozadinu, ink 1 za srednji plan te ink 2 i 3 za prednji plan. To se postiže na 4 načina, gdje u 3 slučaja novi ink „reagira" sa starim:

0. (Force mod) — novi ink prekriva stari
1. (XOR mod) — novi ink se odabire na način „ekskluzivno ILI" prema starom inku, a prikaz ovisi o rezultirajućem inku.
2. (AND mod) — novi ink se „zbraja" sa starim inkom.
3. (OR mod) — novi ink se odabire na „ILI" način prema starom inku.

Detaljnije objašnjenje o AND,OR,XOR se nalazi u četvrtom poglavlju priručnika koji se dobije uz računalo. Razrada planova neposredno ovisi o grafičkim modovima, odnosno o broju boja koje je moguće koristiti za prikaz.

Upravljanje kretanjem

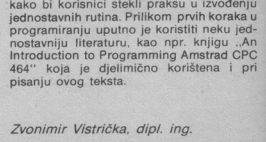
Kretanjem na ekranu se može upravljati kursorskim tipkama ili joystickom. Upravljanje joystickom se može prikazati primjerom programa koji predstavlja igru kojoj se obruč mora provesti preko savinute žice ne dodirujući je Primjer programa:

```
 15 MODE 0
 25 DIM y%(8)
 35 GOSUB 215
 40 REM iscrtavanje screena
 45 GOSUB 265
 55 v=TIME
 65 WHILE  px%<63@
 75 REM pokretanje
 85 GOSUB 485'
 95 REM vrijeme na displeju
115 GOSUB 575
125 WEND
135 REM nova igra?
145 GOSUB 615
155 IF LOWER$ (a$)="d" THEN GOTO
45
165 GOSUB 675
175 IF LOWER$ (a$)="d" THEN GOTO
35
185 MODE 1 : CALL & BBFF : PAPER 0 :
PEN 1
195 END
215 REM koordinate screena
225 FOR b=0 TO 8
235 y%(b)=RND*2@@+1@@
245 NEXT b
255 RETURN
265 MOVE 0,0
275 PAPER 8 : BORDER 3 : CLS
285 DRAWR 0,399,3
295 DRAWR 639,0
315 DRAWR 0,-399
325 DRAWR -639,0
335 MOVE 0,y%(0)+2
345 b=0
355FOR x%=79 TO 639 STEP 80
365 DRAW x%,x%(b)+20
375 b=b+1
385 NEXT x%
395 b=0
415 MOVE 0,Y%(0)-20
425 FOR x%=79 TO 639 STEP 80
435 DRAW x%,y%(b)-20
455 NEXT x%
465 px%=1@ : py%=y%(0)
475 RETURN
485 REM rutina za kretanje
495 a%=px% : d%=py%
515 px%=px%+4 (JOY (0)=8)-4*
(JOY(0)=4)
525 py%=py%+2* (JOY(0)=1)-2
(JOY(0)=2)
535 IF TEST (px%,px%)=3 THEN
px%=a%
545 py%=d% : v=v-3@@ : SOUND
1,1@@,2,0
555 PLOT px%,py%,1
565 RETURN
575 LOCATE 2,2
585 PRINT USING
",###,##";(TIME-v)/3@@
595 RETURN
615 LOCATE 2,4
625 PRINT „Ponovo? (d/n)"
635 a$=LOWER$(INKEY$): IF a$<> „d"
AND a$<> „n" THEN 635
645 LOCATE 2,4
655 PRINT SPACE$(16)
665 RETURN
675 LOCATE 2,4
685 PRINT „Novi screen? (d/n)"
695 a$=LOWER$(INKEY$) : IF a$<>„d"
AND a$<>„n" THEN 695
715 LOCATE 2,4
725 PRINT SPACE$(16)
750 RETURN
```

Program ujedno demonstrira TEST instrukciju koja vraća vrijednost „inka" na označenu točku na ekranu. Potprogram od linije 215 postavlja koordinate paralelnih linija koristeći instrukciju RND. Od linije 265 iscrtava se ekranski prikaz (screen). Linije 515 i 525 očitavaju džojstik pri čijem kretanju dijagonale nisu predviđene. Može se dogoditi da se džojstici nekih proizvođača ne ponašaju kako treba s ovim programom. U tom slučaju je potrebno promijeniti predznake + i - u linijama 515 i 525. Instrukcija TIME se koristi za odredivanje vremena „screena". Varijabla v je postavljena na vrijednost internog sata u liniji 55. Linija 585 proračunava proteklo vrijeme u sekundama oduzimanjem vrijednosti varijable v od TIME, te dijeleći sa 300 (jer interni sat povećava vrijednost svaku 1/300 sekunde). Za svaki kontakt „obruča sa žicom" oduzima se 300 od vrijednosti v.Linja 585 također koristi instrukcije PRINT USING — za prikaz 3 broja ispred i 2 iza decimalne točke.

Ovim primjerima programa ilustrirana je osnovna primjena grafike (prozori, kretanje, animacija, oblici) kao i osnovna konstrukcija igara u Locomotive bejziku. Preporučljivo je s ovim programima eksperimentirati kako bi korisnici stekli praksu u izvođenju jednostavnih rutina. Prilikom prvih koraka u programiranju uputno je koristiti neku jednostavniju literaturu, kao npr. knjigu „An Introduction to Programming Amstrad CPC 464" koja je djelimično korištena i pri pisanju ovog teksta.

Zvonimir Vistrička, dipl. ing.



u nevolji

Ohlađeni „Spektrum"

Mnogo čitalaca nam se javlja sa pitanjem „zašto mi se „spektrum" toliko greje"? To naročito važi za one koji su svoju mašinu opremili mikrodrajvom i koji bukvalno kažu da su našli na novo mesto za kuvanje kafe.

Ne greje se ceo „spektrum", to se sigurno i sami primetili, greje se samo stabilizator napona, i to je normalno. Da se taj stabilizator napona ne bi raspao od sopstvene toplote ljudi su izmislili metalni hladnjak koji odvodi toplotu. Nevolja sa „spekturmom" je u tome što je hladnjak isuviše mali da hladi, on zagreva ceo kompjuter i time mu skraćuje radni vek. Pošto će novi hladnjak zbog svojih dimenzija morati da se nalazi izvan „spektruma-ve" kutije, moraćete da odlemite stabilizator napona i da ga za štampano kolo spojite sa tri žice. Da ne bi došlo do pada napona, žice treba da budu malo debље (1.5 mm) i ne duže od 5%-tak cm. Pošto ste to uradili, savetujemo vam još dve prepravke: stavite jedan mali pljosnati hladnjak na ULA čip i montirajte RESET taster (jednostavno, kedan taster spojite sa izvodima OV i RESET na konektoru). Ako to uradite, budite sigurni da vam se „spektrum" neće uskoro pokvariti.

V.K.

Nevolje sa Paskalom

Nebojša Pantelić se javio zbog problema koji ima sa HISOFT-ovim paskalom. Kaže da mu vrlo često mašina „poludi" dok lista ili kompajlira program.

Takvi problemi su izuzetno česti ipo pravilu se dešavaju ako nedje u programu znakove manje-ili-jednako. veće-ili jednako. ili različito kucate sa SYMBOL/Q, SYMBOL/E ili SYMBOL/W. Te znakove morate da kucate iz dva dela. Ako poste ovog i dalje imate problema, a verujemo da nećete. ostaje vam da nabavite novu verziju paskala, jer vaš... u tom slučaju. ima neke greške.

V.K.

Mašinac na „spektrumu"

Željko Puzović iz Kraljeva pita kako da se iz mašinskog programa vrati na povoljnu bejzik naredbu.

Postoji više načina da se ovo uradi. a na vama je da odlučite koji vam od tih načna najviše odgovara.

Ukoliko se mašinski program pozove sa **RANDOMIZE USR**. . .,, što je najrasprostranjenije i najsigurnije. onda se povratak u bejzik normalno obavlja na naredbu koja inače sledi. Međutim, sasvim je moguće. neposredno ispred **RET**. promeniti sadržaj sistemskih promenljivih NEWPPC i NSPCC. i tako „prevariti" bejzik interpreter. Na primer. ako želimo da se vratimo na treću naredbu po redu u linij 2070. stavićemo na kraju našeg mašinskog programa:

```
LD HL,2070
LD (23618),HL
LD A,3
LD (23620),A
RET
```

Ako se povratak u bejzik vrši na početak neke programske linije, onda je stvar jednostavnija. Moguće je. recimo. ubaciti pre povratka neku fiksiranu vrednost u registar BC. aonda. na osnovu te vrednosti. obaviti skok na uobičajen način iz bejzika:

```
IF USR 23296=7 THEN GO TO
1040
```

Međutim, ovakvo rešenje je često rasipanje memorije i vremena. Isti efekt se dalеko elegantnije postiže ako se mašinski program pozove sa **GO TO USR 23296**, a na povratku, ispred **RET**. stavi **LD BC,1040.** Čak možemo imati i više različitih povratka. na različita mesta u bejzik programu. Samo u BC treba staviti linijski broj na koji će se preneti interpretažija bejzika.

J.S.

Izmene u ROM-u

Momir Stepanović iz Beograda želi da izmeni „spektrumov" ROM, pa ga interesuje da li je moguće prilagoditi operacije kalkulatora tako da se sve, bez izuzetka, pozivaju na uobičajen način, bez korišćenja registra B.

To se može izvesti bez ikakvih problema. Operacije poređenja brojeva i nizova, kao i operacije vređovanja nizova (**VAL** i **VAL$**) imaju taj nedostatak što se u okviru kalkulatora ne mogu pozivati jednostavno preko svog koda. Naime, izvršenje tih operacija počinje mašinskom naredbom **LD A,B**, čime se sam kôd uzima iz registra B i smešta u akumulator radi dalje obrade, tj. prepoznavanja konkretne operacije. Dakle, registar B mora unapred biti pripremljen, kao što smo to već objasnili u dodatku „Sve spektrumove rutine" Pitanje ovde glasi: da li se kôd operacije može dovesti u akumulator na neki drugi način?

Jedna povrsna analiza rutine CALCULATE (&335B) pokazuje nam da se traženi kôd već nalazi u akumulatoru u trenutku poziva operacije, samo što je pri tome još zajedno i naredba **RLCA** na adresi &338C. Prema tome, **LD A,B** uopšte nije potrebno. Dovoljno je, umesto toga, rotirati sadržaj akumulatora udesno sa **RRCA**, to znači da umesto &7B na adresama &353B i &35E2 treba da stoji &0F. Posle ove male izmene, dodatne operacije kalkulatora postaće ravnoprvne sa ostalima.

J.S.

BBC/Electron

Kako snimiti ROM

Mihajla Jovanovića iz Šapca interesuje mogućnost snimanja nekog od pejdžovanih ROM-ova BBC-ja na traku odnosno disk.

Ukoliko želite da snimate na traku, stvar je prilično jednostavna: aktivirajte taj ROM (ako se, na primer, radi o tekst procesoru VIEW, otkucajte *WORD) i iz njegovog komandnog moda otkucajte *SAVE ime 8000 +4000 (umesto 4000 koristite 2000 ako se radi o ROM-u od 8K). Na ovaj način se, na žalost, ne može snimati na disk, jer će uvek biti snimljen ROM sa DFS-om koji se pejdžuje pre bilo koje operacije sa diskom. Metod, osim toga, ne može da se primeni na ROM-ove tipa Toolkita koji nemaju ulaznu tačku za jezike (tzv. *language entry point).*

Univerzalno rešenje je program sa slike. Otkucajte ga, startujte i otkucajte broj podnožja u kojem je ROM ugrađen (između 12 i 15 na standardnom BBC-ju, odnosno između 0 i 15 na BBC-ju koji je proširen ROM - tabloom). Sadržaj ROM-a će biti upisan u RAM počevši od adrese &3000 pa ga možete snimiti uobičajenim *SAVE.

D.R.

```
10 REM
20 REM Prepisivanje sistemsvs ROM-a
30 REM  u RAM počevši od &2000
40 REM
50 REM       Dejan Ristanović 1985.
60 REM
70 MODE 7
80 INPUT TAB(2,5)"Otkucaj broj ROM-a
(0-15) ";N
90 IF N<0 OR N>15 THEN 70
100 INPUT TAB(2,10)"Visina je duzina R
OM-a (0 111 16 K) ";D%
110 IF LEFT$(D%,1)="&" THEN X=EVAL(D%)
ELSE X=D%*1024
120 IF LEFT$(D%,2)="16" THEN X=&4000
130
140 GOTO 70
150 P%=&7000
160 PORT%=&B
170 LDA &FA:PHA:LDA #N:STA &F4:STA &3
80 LDY #0:STY &30:STY &32
190 LDA #&80:STA &31
200 .petlja
210 LDA (&30),Y:STA (&32),Y
220 INC &30:INC &32:BNE petlja
230 INC &31:INC &33
240 LDA &33:CMP #X:BNE petlja
250 PLA:STA &F4:PHA
260 RTS
270 ]
280 CALL &7000
290 PRINT TAB(2,20) "Prenos uspesno za
vrsen!":PRINT
```

Više memorije

Radovan Veljković iz Kotora smatra da šest memorijskih lokacija (mem-0 . . .mem-5). koliko sadrži „spektrumov" kalkulator, nije dovoljan prostor i pita kako da to prevazide.

Istina je da se gotovo svuda piše samo o šest memorijskih lokacija, u koje se upisuje pomoću naredbi &CD...&C5, a iz kojih se čita pomoću &ED . . .&C5. Ali jedini razlog ovom ograničenju jeste činjenica da je za potrebe kalkulatora odvojeno samo 30 bajtova u prostoru sistemskih promenljivih (varijabla MEMBOT). Inače. sasvim lepo rade i naredbe &C6, &C7 itd. za komunikaciju sa memorijama mem-6, mem-7 . . . samo što će u tom slučaju biti uništeni sadržaji sistemskih promenljivih RAMTOP i P–RAMT. kao i baferi za mikrodrajv i kanalske informacije — jer su upravo to bajtovi koji dolaze odmah iza predviđenih 30 u okviru MEMBOT.

Rešenje je prosto: treba otvoriti potpuno nov memorijski prostor na nekom prikladnijem mestu, recimo na adresi 50000, ili bilo gde drugde. Samo treba obavestiti računar o tome da je memorija preměštena, postavljanjem varijable MEM:

```
LD HL,50000
LD (23656),HL
```

Po izvršenju ove dve instrukcije. pri svakom pozivu kalkultaora na raspolaganju nam stoje čak 32 memorijske lokacije. sa oznakama mem -0 mem -1F, i rezervisanim prostorom između adresa 50000 i 50159. Naredbe za upisivanje su &CD...&DF, a za čitanje &ED...&FF.

J.S.

„Galaksija"

Inicijalizacija programa

Bogdan Trifunović iz Beograda ima problema sa programima DIM-/MAT i „Ekranski editor" koji su emitovani preko „Ventilatora" pre skoro godinu dana: sve lepo radi samo za sebe ali se problemi javljaju čim inicijalizaciju ovih programa treba izvršiti iz nekog drugog bejzik programa. Autor oba programa i ovoga teksta priznaje da je isključivi krivac za probleme i da slične prestupe neće ponavljati: u oba programa je ugrađena zaštita (na „Ekranskom editoru" prilično komplikovana) koja otežava izmenu programa u cilju brisanja ili zamene imena autora. Problem sa „Ekranskim editorom" rešavate ako otkucate nezaštićenu verziju iz „Računara 4", dok se za DIM i MAT morate snalaziti na neki drugi način: programi su. na primer, ugrađeni u igru 'Ziul' koja ih uspešno inicijalizuje; pogledajte kako!

(D.R.)

mašinska veza

Bez želje za polemikom jedno je sigurno: jedan od bitnih momenata u donošenju odluke je i činjenica da baš taj računar imamo na raspolaganju i da baš taj programski jezik znamo. U ogromnom broju slučajeva to su „spektrum" i bejzik. Da li to znači da veliki numerički problem treba da se ostavi za neka sretnija vremena, ili da neko preobimnije sortiranje treba, naprosto, zaboraviti. Svakako ne! Naravno, neozbiljno bi bilo da poslove koji bi preznojili i velike računare prepuštamo „igračkama", ali i još neozbiljnije da arčimo vreme skupih sistema i da na njima u nedogled proveravamo svoje zamisli, testiramo algoritme i radimo sve ono što se, po prirodi stvari, radi ugodnije i konfornije u tišini svoga stana. Ako je već tako, proizilazi da ni bejzik nije tako loš programski jezik i da, iako nije baš zasnovan na savremenim principima programske nauke, može veoma dobro da posluži mnogim zadacima. Interaktivnost je njegovo najjače oružje — mogućnost intervencije u svakom trenutku pruža mnoge pogodnosti. Kada se ideje provere, dobiju test rezultati i optimizuju algoritmi, nije nikakav problem preći na bilo koji drugi jezik, koristeći sve njegove dobre strane (o ovome se čak može voditi računa i od samog početka).

Znači, to je interaktivni softver!

Zajedno su jači

Osnovna zamerka bejziku „spektruma" je njegova sporost, a prevodioci za bejzik su u najvećem broju slučajeva takvi da u tom domenu ne daju velika poboljšanja, posebno kada je problem numeričke prirode. Drugi nedostatak (ali ne samo bejzika) je činjenica da mnoge sistemske rutine ne rade kako bi mi to želeli, a nije redak slučaj ni da nailazimo na greške autora sistemskih programa. Ali, ako želimo da se neka trigonometrijska funkcija, ili čitava jednačina računa po nekom našem algoritmu, neophodno je da program napišemo u mašinskom jeziku i da ga povežemo sa bejzikom. Kod nekih računara, to je već obezbeđeno, ali je kod „spektruma" potrebno, pre svega, imati program za povezivanje.

Kada mašinski program nema ulazno-izlaznih veličina, problem, praktično ne postoji, ali ako je mašinskom delu programa potrebno preneti vrednosti nekoliko promenljivih i iz njega rezultat vratiti nazad u bejzik, i da pri tom radimo sa realnim brojevima, onda se mora dopisati mašinski program koji to omogućuje. Uzmimo, na primer, da je potrebno sumirati neki rad. FOR-NEXT petlja na „spektrumu" radi relativno sporo, gubi se vreme na interpretaciju i ima rezona da takav posao, pogotovu ako se u nekom programu često obavlja, prepustimo mašincu. Za ubrzavanje programa često je dovoljna samo mala intervencija na kritičnom mestu i nije neophodno sve pisati u nepreglednim kodovima mašinskog jezika. Uostalom, staro je pravilo da je lanac onoliko jak koliko je jaka njegova najslabija karika. Tako je i program onoliko brz koliko je brz njegov najsporiji deo.

U prilogu br. 1 dat je polazni bejzik program koji ćemo kasnije transformisati. Taj program računa šestostruki zbir prvih 1000 članova reda čiji je opšti član:

$$A_n = 1/n^2$$

Sumiranje ovog reda treba, naravno, obaviti počev od najmanjeg člana, jer bi se, u suprotnom, dobio mnogo netačniji rezultat. Sabiranje malih brojeva sa velikim „istiskuje" cifre rezultata i gubi se na tačnosti. Zato indeks petlje počinje od 1000 i opada.

Rezultat programa (parcijalni zbir) iz priloga br. 1 trebalo bi, kada bi se uzeo znatno veći broj sabiraka, da bude jednak kvadratu broja PI. Na žalost, hiljadu članova dovoljno je za samo grub rezultat i očigledno je da za računanje kvadrata broja PI ne možemo koristiti ovakav algoritam. Ali to nije razlog da odustanemo od primera. Ako program unesete u vaš „spektrum" i izvršite ga, dobićete rezultat posle 15 sekundi. Može li to istim algoritmom brže? Može, evo kako.

Prenošenje vrednosti

Da bismo rešili problem prenosa argumenata i rezultata iz bejzika u mašinac i obrnuto, treba najpre da se podsetimo kako „spektrum" beleži promenljive. Sistemska promenljiva VARS (vrednost na adresi 23627 i 23628) sadrži adresu memorije od koje se smeštaju promenljive programa koji se nalazi u računaru. Na prvoj adresi, koju pokazuje VARS, nalazi se prvo slovo imena prve promenljive, zatim sledeće slovo i tako do poslednjeg, a potom pet bajtova zauzima sama vrednost. Kada naredimo: PRINT X, računar u zoni promenljivih traži „X" i, kada ga pronađe, prikazuje decimalni prevod petobajtnog zapisa koji iza oznake „X" sledi u memoriji. Redosled promenljivih je onakav kako se u programu pojavljuju. Zapis, a i mesto na kome se pamte stringovi, elementi nizova i matrica i indeksi petlji, nešto je drugačije rešen i ovde o njima neće biti reči. Detalji se mogu naći u umetku „Sve spektrumove rutine".

Dakle, ako u programu u prvoj liniji napišemo X=0 (sama brojna vrednost nije od značaja za njen položaj u memoriji), tada se PRINT PEEK(23627)+256*PEEK (23628) dobijamo adresu u memoriji na kojoj se nalazi početak promenljive X. Neka je ta adresa ADR. Vrednost promenljive X biće, ma kako da nastavimo program, uvek u pet bajtova počev od ADR+1. Ako, kao sledeću naredbu napišemo Y=0, onda će se stalno na pet bajtova, počev od ADR+7, nalaziti vrednost druge promenljive.

Dok jedni smatraju da nisu ništa uradili ako u toku dana ne napišu bar nekoliko „kila mašinca", drugima je ispod časti da pišu programe na bilo kom jeziku osim na prologu (ili bar paskalu). Neko će reći: sve je to jedno isto. Ipak, da li se svaki problem može rešiti na svakom jeziku na jednako dobar način? Kako mentalitet programera utiče na izbor „oruzja"? Koliko sam problem odreduje primenu ovog ili onog prevodioca, ovog ili onog interpretera? Povezivanje mašinskih rutina i bejzik programa često može biti veoma korisno. Kada je potrebno da „mašinac" nešto računa, a da argumente računa možemo lako da kontrolišemo i da se rezultat računa vraća u bejzik, tada treba rešiti probleme prenosa. Malo mašinskog koda, kao i obično, lako rešava problem.

```
Prilog br.1: TEST PROGRAM

10 Y=0
20 FOR N=1000 TO 1 STEP-1
30 Y=Y+1/(N*N)
40 NEXT N
50 Y=6*Y:PRINT Y
```

Prilikom unošenja u računar primera 1 i 3 treba imati na umu da „spektrum" u naredbama dodele zahteva službenu reč LET pa zato, umesto Y=0, treba pisati LET Y=0

```
Prilog br.2:Prenosenje argumenata iz
            bejzika u masinski deo i
            rezultata nazad u bejzik
D6DB                ORG  55000
                2 ;
                3 ;PRVA PROMENLJIVA
                  NA STEK
                4 PNS
D6DB 2A4B5C     5       LD   HL,(236
23)
D6DE 23         6       INC  HL
D6DF CDB43b     7       CALL #33B4
D6E2 C9         8       RET
                9 ;
               10 ;DRUGA PROMENLJIVA
                  NA STEK
               11 DNS
D6E6 2A4B5C    12       LD   HL,(236
23)
D6E9 010700    13       LD   BC,7
D6EC 09        14       ADD  HL,BC
D6ED CDB433    15       CALL #33B4
D6F0 C9        16       RET
               17 ;
               18 ;SA STEKA U TRECU
                  PROMENLJIVU
               19 SST
D6F1 CD8F35    20       CALL #35BF
D6F4 EB        21       EX   DE,HL
D6F5 2A4B5C    22       LD   HL,(236
23)
D6F8 010800    23       LD   BC,13
D6FB 09        24       ADD  HL,BC
D6FC EB        25       EX   DE,HL
D6FD 010500    26       LD   BC,5
D700 EDB0      27       LDIR
D702 C9        28       RET
```

Sva računanja „spektrum" obavlja pomoću kalkulatora sa pokretnim zarezom u koji se ulazi sa RST&28, a čije se funkcije koriste prosto redanjem njihovih šifri. Sve operacije definisane su nad brojem koji se nalazi na vrhu steka kalkulkulatora, pa je potrebno da promenljivu iz bejzika, pre nego što pozovemo kalkulator, postavimo na vrh njegovog steka. Na steku važi uobičajeno pravilo: poslednji unutra — prvi napolje. U ovakvim okolnostima najjednostavnije prenošenje vrednosti promenljive na stek kalkulatora sastoji se u korišćenju rutine iz ROM-a & 33B4, koja pet bajtova počev od adrese koja se nalazi u HL registar paru smešta na vrh steka kalkulatora. Ona, usput, radi još mnogo neophodnih stvari, kao što je testiranje da li na steku ima još mesta za nove podatke, gde se nalazi vrh steka, itd). Suprotan postupak, prenošenje vrednosti sa vrha steka kalkulatora u zonu promenljivih, obavlja se preskikavanjem (LDIR) pet bajtova, pri čemu se u DE registar paru nalazi adresa na koju se preslikava, a u HL, adresa vrha steka kalkulatora. Ovaj podatak daje rutina iz ROM-a na & 35BF, a adresu memorije u koju podatak prenosimo dobijamo uvidom u sadržaj sistemske promenljive VARS i dodavanjem odgovarajućeg broja bajtova. Ako su sve promenljive, iii bar prve tri, jednosIovne, tada treba dodati 1 za prvu, 7 za drugu, 13 za treću itd.

```
Prilog br.3: TEST PROGRAM
             (bejzik deo)

10 N=1000:A=6:Y=0
20 RANDOMIZE USR 50000
30 PRINT Y
```

```
Prilog br.4: TEST PROGRAM
             i MAC DEO PROGRAMA
                1 ;
                2 ;
C350            3       ORG  50000
C350 CD50C3     4       CALL DNS
C353 CD53C3     5       CALL PNS
C356 EF         6       RST  #28
C357 C402       7       DEFB #C4,2
C359 A001       8       DEFB #A0,1
C35B C531       9       DEFB #C5,49
C35D 04A1       10      DEFB 4,#A1
C35F 0185       11      DEFB 1,5
C361 10E5       12      DEFB 16,#E5
C363 A103       13      DEFB #A1,3
C365 31         14      DEFB #31
C366 3700       15      DEFB #37,0
C368 F302       16      DEFB -13,2
C36A E404       17      DEFB #E4,4
C36C 38         18      DEFB #38
C36D CD6DC3     19      CALL SS3
C370 C9         20      RET
```

Program je nezavisan od mesta učitavanja. Jedino treba zapamtiti da, ako ga unesemo na adresu xxxxx, tada sa CALL xxxxx (CALL PNS) dobijamo vrednost prve promenljive iz bejzika na vrhu steka kalkulatora, sa CALL xxxxx+8 (CALL DNS) dobijamo vrednost druge promenljive iz bejzika na vrhu steka, a sa CALL xxxxx+19 (CALL SST) sa vrha steka kalkulatora vrednost prepisujemo u trećeimenovanu promenljivu bejzika (uz uslov da su sve tri jednoslovnog imena). Uz male modifikacije ovi bi se programi mogli višestruko proširiti i uskladiti sa drugačijim zahtevima.

Primer iz prilOga br. 1 može se ubrzati više od dva puta ako se sumiranje obavi u mašinskom jeziku. U bejziku bi ostalo veoma malo: postavljanje polaznih vrednosti, poziv mašinskog programa i štampanje rezultata. Naravno da se i ovo moglo poveriti mašincu, ali tada bi za svaku promenu morali da menjamo program, a to nije cilj. Deo programa u bejziku izgleda kao u prilogu br. 3. Ovde je bitno zapamtiti da moramo zauzeti početna mesta za prve tri promenljive, od kojih prve dve predstavljaju ulazne veličine za mašinski deo programa, a treća je izlazna vrednost. Adresa mašinskog potprograma je proizvoljno odabrana i može se po želji menjati.

Asemblerski listing mašinske sekvence dat je u prilogu br. 4. Za njeno razumevanje neophodno je poznavanje „spektrumovog" kalkulatora i ovde se na tome nećemo zadržavati. Ipak treba reći da se pri korišćenju kalkulatora mora dobro paziti koje interne memorije on koristi kod pojedinih funkcija. Na primer, kod računanja korena kalkulator koristi MO i MI i ako se u njih nešto spremi, pa zatim pozove računanje korena, posle toga ćemo biti iznenađeni sadržajem koji tamo nađemo. Korisno bi bilo za svaku kalkulatorsku funkciju zapisati koje interne memorije koristi i njih izbegavati. Ovo se iz disasembliga „spektrumovog" ROM-a jasno vidi.

Traganje za kraćim putevima može da bude veoma zanimljivo i pruža mnoge mogućnosti. Na opisan način „spektrum" bi se mogao naučiti da računa brže. Tačnije iii. čak, da se promeni način zapisivanja pojedinih promenljivih i tako proširi opseg i broj značajnih cifara do željenih granica. Nije problem dodati bejziku „spektruma" funkcije koje nema, a i dalje ostajemo u interaktivnom jeziku u kome u svakom trenutku možemo da menjamo ono što smatramo da je potrebno. Uz izvesnu doradu, mogla bi se promeniti sintaksa i formirati naredba u bejziku tipa CALL(45543:×1,×2,...,y1,y2...) koja bi mašinskom programu na adresi 45543 kao ulazne argumente slala vrednosti promenljivih x1, x2, ..., a iz nje se u bejzik rezultati smeštali u promenljive y1, y2. ... Da vam ovo možda ne liči na fortran?

Veza sa „mašincem"

Ovaj postupak realizovan je u asemblerskom programu datom u prilogu br. 2.

Ninoslav Čabrić

Škola logičnih igara (1)

volite li paskal?

Paskal program se sastoji od nekoliko sekcija. Prva od njih identifikuje program i sastoji se od jedne jedine naredbe, na primer:

```
program proba (input, output);
```

Ovom naredbom smo naglasili da je ime programa koji pišemo proba i da on ima ulazne i izlazne veličine; u programu će se, dakle, koristiti naredbe read i write. Simbol ";" na kraju reda je uobičajeni separator instrukcija na paskalu; obaveza da se ovaj simbol stavi između svake dve naredbe je omogućila da se bilo koja konstrukcija proteže u više redova: mogli smo, da smo smatrali da to ima nekog smisla, napisati prethodni red i kao:

```
program
proba (inp
ut, output);
```

Deklaracija promenljivih

Iza imena programa nailazi odeljak za deklarisanje konstanti i promenljivih. *Svaka* promenljiva koja se koristi u okviru programa *mora* da se deklariše, pri čemu se navodi njeno ime i tip. Osnovni tipovi promenljivih su celobrojne (integer), racionalne (real), logičke (Boolean) i alfabetske (char). Tako se iza reda *program proba (input, output)* mogla pojaviti konstrukcija:

```
const pi =3.14159
var i,j,: integer;
ri, r2:real;
flag:Boolean;
slovo:char;
```

U programu koji će slediti reč 'pi' je ekvivalentna zamena za broj 3.14159; 'i' i 'j' su celobrojne, 'ri' i 'r2' racionalne, 'flag' je logička, a 'slovo' alfanumeričke promenljive (u paskalu alfanumeričke promenljive mogu da „zapamte" samo po jedno slovo tako da bi *slovo=ABCD'* bilo ekvivalentno sa *slovo;:='A').*

Lepa osobina paskala je što nismo ograničeni samo na osnovne skalarne tipove. Možemo, na primer, da uvedemo promenljivu tipa *boje* koja će imati vrednosti belo, crno, zeleno ili žuto(primetimo da *bi:=crno* nije ekvivalent bezizk naredbe *bi=„crno"* jer se u drugom slučaju promenljivoj bi dodeljuju slova *c, r, n* i *o*, dok se u prvom promenljivoj bi dodeljuje vrednost *crno* koja je jednako dobra kao i svaki drugi broj). Za definisanje specijalnih tipova je odgovorna *type* sekcija koja se nalazi između *const* i *var*. Na primer:

```
type niz = array (0..100) of real;
boje = (belo, crno, zeleno, žuto);

var A:niz;
    b1, b2, b3: boje;
    i, j, k:integer;
```

Ovom smo sekcijom definisali 7 promenljivih. A je niz koji bi se u bejziku dobio primenom naredbe DIM A(100). Primetimo, međutim, da se indeksi bilo kog niza u bejziku kreću između 0 i nekog *n* dok smo u prethodnom primeru mogli da napišemo *niz=array [50..150] of real* čime bi se indeksi niza A kretali između 50 i 150, a ne između 0 i 100. Što se tipa 'boje' tiče, u glavnom programu ćemo moći da pišemo *b1:=crno* ili *if b1=b2 then . . .* ali će naredba *b3:=ljubičasto* izazvati grešku jer nismo predvideli da je ljubičasta jedna od boja. Posebno je zgodno što deklaracijom *type boje* nismo samo predvideli imena boja već i njihov redosled: uz gornju deklaraciju je belo „najmanja" a žuto „najveća" boja tako da bi *if žuto zeleno* dalo odgovor *true*.

Procedure i funkcije

Iza dekleracija konstanti, tipova i promenljivih (nismo pomenuli mogućnost definisanja labela, jer u paskalu nećemo koristiti naredbu goto niti mogućnost definisanja složenih struktura podataka kao što su skupovi i slogovi) sledе procedure i funkcije. Svaka procedura ili funkcija je sama za sebe mali paskal program koji ima zaglavlje, argumente, deklaracije, definicije svojih funkcija i, naravno, izvršni deo. Na primer:

```
slika 1f

ISO-Pascal compiler V- R1.00

  1 0 - program faktorijel(input,output);
  2 0 - var n,f:integer;
  3 0 -
  4 0 - procedure fakt(n:integer;var f:integer);
  5 1 - var i:integer;
  6 1 - begin
  7 1 -     f:=1;
  8 1 -     for i:=1 to n do f:=f*i;
  9 1 - end;
 10 1 -
 11 1 -
 12 0 - begin
 13 0 -     write('Unesi n '); readln(n);
 14 0 -     writeln(n,'! = ',fakt(n));
 15 0 - end.
 16 0 -
 17 0 -
0 Compilation error(s)
Code size = 164 bytes
```

Na slici 1 je prikazan kratak paskal program koji izračunava faktorijel nekog broja. Radni deo posla obavlja procedura *fakt* koja ima ulaznu celobrojnu veličinu *n* i izlaznu veličinu *f* istog tipa. Primetimo da se, u okviru navođenja argumenata procedure, sve veličine koje će u toku njenog izvršavanja menjati vrednost moraju označiti kao *var*; da smo izostavili ova tri slova procedura ne bi mogla da promeni vrednost jednog od argumenata, pa tako ni da dojavi glavnom programu rezultate svoga rada!

Procedura *fakt* ima lokalnu promenljivu i. Ukoliko niste sasvim sigurni šta su lokalne promenljive, pogledajte tekst „Funkcije, potprogrami, procedure" u „Računarima 7"; ukratko, vrednosti lokalne promenljive važe samo u toku izvršavanja same procedure, dok se izvršavanjem naredbe *i:=1* u proceduri ne bi promenila vrednost promenljive i koja je definisana u okviru glavnog programa.

```
slika 2f

ISO-Pascal compiler V- R1.00

  1 0 - program faktorijel(input,output);
  2 0 - var n,f:integer;
  3 0 -
  4 0 - function fakt(n:integer):integer;
  5 1 - var i,f:integer;
  6 1 - begin
  7 1 -     f:=1;
  8 1 -     for i:=1 to n do f:=f*i;
  9 1 -     fakt:=f
 10 1 - end;
 11 1 -
 12 0 - begin
 13 0 -     write('Unesi n '); readln(n);
 14 0 -     writeln(n,'! = ',fakt(n));
 15 0 - end.
 16 0 -
 17 0 -
0 Compilation error(s)
Code size = 164 bytes
```

Umesto procedure, za izračunavanje faktorijela smo mogli da koristimo funkciju kao na slici 2. Funkcija u principu ima više ulaznih veličina i samo jednu izlaznu koja se dodeljuje imenu same funkcije. Pošto je poslednja naredba funkcije *fakt* upravo *fakt:=f*, naredba glavnog programa *write (fakt(n)* će na ekranu ispisati vrednost faktorijela broja *n*. Da bi paskal kompajler mogao da proveri ispravnost ovoga ispisivanja, kao i ispravnost naredbi tipa *rezultat:=fakt (n)*, neophodno je pri deklaraciji funkcije napisati i tip rezultata koji ona vraća; otuda ono *function fact(. . .) : integer.*

Izvršni deo

U paskalu postoji jedna na prvi pogled čudna konvencija: bilo koja procedura, funkcija, petlja ili neka druga struktura, uključujući i sam glavni program, sastoji se od jedne jedine naredbe. Jedna naredba je, jasno, sasvim nedovoljna da bi se bilo šta obavilo, pa se umesto neke izvršne rečenice koristi konstrukcija *begin . . . end*. Kada paskal kompajler analizira neku strukturu i očekuje izvršnu naredbu, pronađeno *begin* će izazvati izvršavanje svih naredbi do odgovarajuće naredbe *end*. U programu sa slike 2 glavni program ima nekoliko izvršnih naredbi koje su međusobno razdvoje-

Čovek, kažu, najbrže nauči da pliva kada ga bacite u vodu. Slično je i sa programskim jezicima — učimo ih tek onda kada su nam potrebni. Primeri programa kojima ćemo propratiti našu školu logičkih igara, s kojom počinjemo od sledećeg broja i sledeći nastavak „Putovanja u središte ROM-a" pisani su na paskalu, jer smo zaključili da standardni bejzik nije dovoljan za jednostavnu ilustraciju tehnika o kojima govorimo; učinilo nam se, osim toga, da će paskal, kao dobro standardizovan jezik, predstavljati prihvatljivo rešenje za vlasnike raznorazniih medusobno nekompatibilnih kompjutera. Cilj naših škola, medutim, nije kucanje gotovih programa već razumevanje principa njihovog rada. Zbog toga ćemo posvetiti pažnju osnovnim osobinama paskala bez ambicija, naravno, da zamenimo daleko kompletniju školu ovoga jezika koju planiramo za jedan od sledećih brojeva „Računara".

ne simbolom ';' i koje su smeštene izmedu *begin* i *end* čineći kontrolnu strukturu koju nazivamo *sekvenca*.

Osim sekvence, za strukturiranu programiranje su potrebni isptivanje i petlja. Ispitivanje se raalizuje primenom konstrukcije *if ... then ... else* koju znamo iz bejzika. Razlika je zaista mala: ukoliko logički iskaz iza *if* ima vrednost *true* biće izvršena *jedna* naredba koja se nalazi iza *then* dok će u slučaju da iskaz ima vrednost false biti izvršena *jedna* naredba iza *else* (else sekcija se, jasno, može i izostaviti). Umesto te naredbe se može naći reč *begin* i, iza nje, čitava sekvenca naredbi (u kojoj smeju da se nalaze i nova ispitivanja) koja se završava sa *end*. Posebno vas upozoravamo na jednu početničku grešku koja može da vam donese mnogo glavobolja:

```
if a=b then; write ('Jednaki su');
```

Simbol ';' koji smo istakli u prethodnoj naredbi je, kao što smo rekli, *separator* instrukcija. On je stavljen iza *then* i, samim tim, predstavlja kraj čitave *if ... then ... else* strukture. Ukoliko je 'a' jednako 'b', biće izvršen *deo te konstrukcije* koji se nalazi iza *then*, što znači da neće biti izvršeno ništa pa se na ekranu *neće* pojaviti željeni tekst 'Jednaki su'!

Paskal podržava tri vrste petlji: *while- ... do, repeat ... until i for ... do*. Prva petlja je univerzalna: iza while se piše neki logički uslov, a iza *do* jedna naredba (ili *begin ... end*) koja se izvršava *sve dok je uslov ispunjen*. Iza *until* se piše logički uslov i računar će izvršavati naredbe izmedu *repeat i until sve dok se taj* uslov ne ispuni. *For ... do petlja je relativno slična FOR-NEXT* strukturi na bejziku, s tim što STEP može da bude samo 1 ili −1. Tako ćemo umesto FOR I=1 TO 100 pisati *for i:=1 to 100 do (naredba)*, a umesto FOR I =100 TO I STEP −1: *for i:=100 downto 1 do (naredba)*. Brojač u petlji može da bude promenljiva bilo kog tipa osim racionalne što znači da bismo mogli da napišemo *for bi¿¿rno to žuto do* ako smo prethodno definisali tip *boje*. Na slici 3 je dat primer funkcije za računanje faktorijela koja koristi svaku od tri pomenute vrste petlji.

Brojni izrazi se u paskalu pišu kao i u bejziku uz sličan prioritet računskih operacija. Za dodeljivanje vrednosti promenljivima se, kao što vidimo, koristi ':=' koje zamenjuje LET u bejziku. Tako je *a:=3* naredba koja dodeljuje vrednost 3 promenljivoj *a*,. dok je *a=3* deo logičkog izraza kojim se ispituje da li je sadržaj promenljive *a* broj 3. Logički izrazi se pišu u sklopu *if* naredbe ili pri računanju sa logičkim promenljivima (tip *Boolean)*. Zanimljivo je da tvorci paskala nisu najsrećnije rešili prioritet logičkih operacija tako da treba pisati *if (a=3) and (b=4) then . . . a nikako if a=3 and b=4 then . . .*

Procedure i funkcije se pozivaju jednostavnim navodenjem njihovog imena i spiska argumenata u zagradi. Tvorci paskala su nam obezbedili nekoliko standardnih funkcija koje su, zajedno sa kratkim opisima delovanja, navedene na slici 4. Od ugradenih procedura posebno su važne *read, readln, write* i *writeln;* iza svake od njih se navodi spisak promenljivih i konstanti koje se nalaze u zagradi i razdvojene su zarezima. Razlika izmedu *write* i *writeln* je u tome što posle izvršavanja prve kurzor ostaje u istom redu (kao da smo u bejziku napisali PRINT „ABCD";) dok druga, kao što joj i ime govori (write line), izaziva prelazak kurzora u sledeći red. Autori paskal kompajlera za razne kućne računare su ovaj jezik opremili dodatnim procesurama za crtanje po ekranu i proizvodenje zvuka koje su opisane u okviru uputstva za upotrebu samog kompajlera.

Ilustracija: Mika Marković

Paskal kompajleri

Paketi koji se nazivaju paskal kompajleri obično se sastoje od editora koji omogućava unošenje izvornog teksta programa i njegovo docnije ispravljanje, kompajlera koji prevodi taj izvorni tekst u mašinski program ili (često) u takozvani „p kod" i (eventualno) *rum time interpretera* koji mora da se nalazi u memoriji da bi se program preveden na *p kod* izvršavao. Najviše će vam truda biti potrebno da ovladate editorom; kada uz njegovu pomoć pripremite program, izdaćete komandu poput *compile* i on će biti preveden a zatim, sa *run* ili *go*, i startovan. Prevodenje i startovanje ćete morati da ponavljate posle svake, pa i najmanje ispravke programa.

Pročitavši ovaj tekst bićete u stanju da razumete rad svih primera koje pripremamo za našu školu logičkih igara, ali ćete i dalje imati problema sa pisanjem samostalnih paskal programa. Najbolje je da pokušate da otkucate program za „Kalah" i da ga prevedete. U kucanju ćete, naravno, zaboraviti da otkucate poneku tačku i zarez, pisati for i:=1 to n, a ne for i:=1 to n *do*, zaboravljati dvotačku ispred jednako i slično. Tražeći ove greške, naučićete osnove paskala i, uz malo dobre volje, početi da pišete samostalne programe. To će biti pravo vreme da se, uz malu pomoć sledećih brojeva „Računara", upoznate sa ostalim svojstvima ovog zanimljivog jezika.

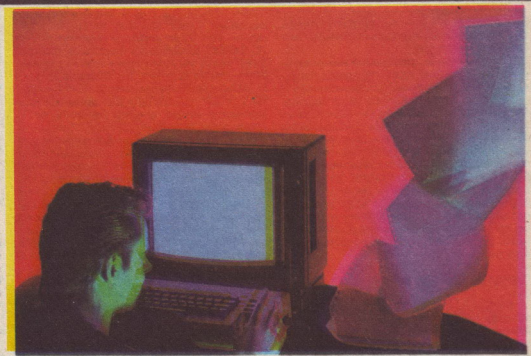
Dejan Ristanović

plavi koverti iz računara

Obrada ličnih dohodaka

Računari u poslovnoj primeni

Izbor obrade ličnih dohodaka za prvu temu naše serije o poslovnoj primeni računara možda i nije najbolji. Međutim, ako imamo u vidu da su gotovo svi veliki računski centri krenuli upravo sa obradom ličnih dohodaka kao prvom postavljenom aplikacijom, nema razloga da ne ukažemo na mogućnost primene kućnog računara baš u toj oblasti. Izbor teme uslovljen je prvenstveno činjenicom da većina čitalaca zna šta su to lični dohoci i da tu nije potrebno preširoko objašnjavanje problema. Dovoljno je da se postavi zahtev, pa da većini bude jasno o čemu se radi.



Projektni zahtev

Izvršiti obradu ličnih dohodaka za radnu organizaciju koja u svom sastavu ima više OOUR-a, sa oko 500 radnika i sa mestima stanovanja u različitim opštinama, pa i republikama. Obradu izvršiti tako da se kao rezultati dobiju:

— izveštaj radniku o obračunu ličnog dohodka — „koverat"
— sve virmanske uplatnice koje se javljaju kao obaveze OOUR-a prema društvu i kreditorima
— sve rekapitulacije virmanskih uplatnica
— analitički izveštaji kreditorima
— statistika ličnih dohodaka (kao izveštaj RAD-1 i slični)
— mogućnost izrade godišnjeg izveštaja za SIZ PIO (izveštaj M-4)

Obradu izvršiti na personalnom računaru sa sledećom konfiguracijom:

— računar sa 64Kb
— dve disketne jedinice maksimalnog kapaciteta po jedinici do 300Kb (poželjno što manje)
— serijski štampač sa oko 100 znakova u sekundi.

Štampa „koverta"

Kao što je već rečeno u prethodnom članku, usko grlo za primenu kućnih računara u radnim organizacijama predstavlja relativno skromna brzina raspoloživih štampača. Stoga ćemo prvo i analizirati vreme potrebno da se obavi potrebna štampa pri obradi ličnih dohodaka.

Štampa odgovarajućeg „koverta" — izveštaja radniku o izvršenom obračunu ličnog dohodka, kao i o izvršenim obustavama predstavlja svakako onaj deo štampe koji će zahtevati i najviše vremena. Kako se ovakav dokument štampa za svakog radnika, to ćemo proceniti vreme potrebno za štampu samo jednog izveštaja. Pri tom treba imati na umu da je poželjno štampu vršiti na unapred pripremljenom (predštampanom) obrascu, kako bi se izbeglo gubljenje vremena na štampi standardnih opisa sadržaja pojedinih podataka. Ovo nije nikakvo ograničenje, već jednostavno zahtev koji se postavlja i pri svakoj ozbiljnijoj obradi na velikim računarima. Ovaj izveštaj treba da sadrži: matični broj radnika, prezime i ime radnika, mesec obrade, broj bodova radnika po rešenju, plansku vrednost boda, procenat ostvarenja, časove rada rasčlanjene prema vrstama, obračunati neto i bruto lični dohodak, minuli rad, sve obustave po njihovim vrstama sudsko-administrativne zabrane, akontacije, kredit, sindikalnu i partijsku članarinu, te iznos za isplatu.

Pored ovih podataka, koji se mogu smatrati obaveznim, poželjno je da se nađu i podaci o svakom obustavljenom kreditu (naziv kreditora i trenutni saldo kredita), kao i pomenično navedeni svi doprinosi koje je radnik platio iz svog ličnog dohotka. Mada ovaj poslednji zahtev predstavlja čak i obavezu prema ZUR-u, samo mali broj radnih organizacija sa velikim sistemima za obradu podataka ima ovaj način izveštavanja, te ga u našem slučaju možemo i zanemariti, jer bi nam štampanje teksta za objašnjenje (na primer „doprinos SIZ-u za penzijsko i invalidsko osiguranje po stopi od . . .") oduzelo izuzetno mnogo vremena, tim pre što ovakvih doprinosa ima poprilično. Ako saberemo prosečan broj znakova koje je potrebno obavezno prikazati dobićemo oko 100 znakova, a ukoliko želimo i prikaz kreditora broj će biti oko 150 (ovde nam na ruku ide i novi propis da se svi iznosi štampaju bez prikazivanja para). Međutim, ovako mali broj znakova nas ne sme zavarati. Naime, ako raspolažemo sa štampačem od 100 znakova/sekundi, to još uvek ne znači da ćemo čitav „koverat" odštampati za 1,5 sekundi.

Podatak koji proizvođači štampača daju je broj znakova u sekundi, jer je on interesantan za međusobna poređenja, i ima svoju punu vrednost u slučaju da se štampaju redovi koji sadrže maksimalan broj znakova koje je moguće smestiti u jedan red. Ukoliko pak red nije pun, a u našem slučaju je pre „prazan" nego pun, za nas je od bitnog značaja vreme potrebno da se papir pomeri za jedan red. Ovaj podatak nije dostupan, ali ga je lako eksperimentalno proveriti na raspoloživom štampaču. Kod pogodno projektovanog „koverta" čitavu štampu je moguće postaviti u 40 kolona raspoređenih u više redova. Na osnovu iskustva, pod ovim uslovom, svaki iole pristojniji štampač sa brzinom od 100 znakova u sekundi može odštampati jedan red, sa skokom na sledeći red, za približno 0,5 sekundi. Kako naš koverat nema potrebe da sadrži više od 30 redova, to je teorijski potrebno 15 sekundi za štampu jednog „koverta". Ovako izračunato vreme treba uvećati još za 30 odsto, te konačno dobijamo da je za štampu jednog koverta potrebno oko 20 sekundi. Računajući sa 500 radnika, vreme potrebno za štampu svih koverata iznosi 10.000 sekundi, ili oko 3 sata.

Virmani u izveštaju

Štampanje virmana je drugo po obimu. Za razliku od prethodnog slučaja kod „ko-

Iako se u poslednje vreme za kućne računare nalazi i u kući poneki ozbiljniji posao, većina i dalje ostaje da čami neiskorišćena, a novopečeni programeri nisu ni svesni kakvo moćno oruđe imaju u rukama. Kućni računari, međutim, idealni su za malu privredu, a mogu, na određenim poslovima, da nadu svoje mesto čak i u najvećim radnim organizacijama, bez obzira da li u njima već postoji računarski sistem. U ovoj seriji tekstova analiziramo neke značajne mogućnosti primene ličnih računara za potrebe organizacija udruženog rada.

verata", ovde smo oslobođeni razmišljanja o podacima koje je potrebno prikazati, kao i o načinu njihovog prikazivanja. Naime, zakonodavac je propisao oblik ovog dokumenta, uključujući obim podataka kao i njihov raspored. Na dokumentu postoji 12 „radnih" redova, dok veličina dokumenta iznosi 24 reda. Maksimalna dužina reda je pri tom 64 znaka. Kako se od 12 raspoloživih redova obično koristi 10, a maksimalna dužina reda je obično 60, to je, teorijski potrebno, oko 7 sekundi, ili praktično oko 10 sekundi za jedan virman. Broj virmana koji je potrebno odštampati zavisi od organizacione strukture radne organizacije, to jest od broja OOUR-a koji se javljaju kao nalogodavci na ovom dokumentu. Može se reći da je za potrebe doprinosa za radnike i pripravnike, sindikalnu i partijsku članarinu i akontacije potrebno od 50 virmana po OOUR-u. Drugi deo virmana koji je potrebno odštampati odnosi se na kreditore kod kojih su radnici uzeli kredite. Na 500 radnika može se računati sa oko 300 različitih kreditora, te je potrebno odštampati isti toliki broj virmana. To ukupno iznosi 350 virmana po jednom OOUR-u, ili 3.500 sekundi, odnosno oko jedan čas rada štampača.

Treći deo štampe odnosi se na izveštaj koji se šalje kreditorima o izvršenim obustavama. Ovi izveštaji obično sadrže red popunjen maksimalno, to jest sadrži 80 znakova u jednom redu, dok je broj redova po izveštaju teško proceniti. Izveštaj obavezno sadrži zaglavlje koje objašnjava tekst koji sledi ispod i obično zahteva 5 redova. Kako je prosečan broj kredita po radniku maksimalno 2, potrebno je odštampati i 1.000 ovih redova, što čini ukupno 2.500 redova. Kako izvestan broj kreditora kreditira samo jednog radnika, ovaj izveštaj nije potrebno za njega štampati, jer se svi podaci mogu naći na virmanu, pa se može očekivati da je potrebno odštampati oko 2.000 redova za potrebe ovog izveštaja. Sa brzinom štampe od 100 znakova u sekundi jedan red je moguće odštampati, računajući i vreme potrebno za skokove na susedni red, teorijski, za oko 1 sekundu, a praktično za oko 1,3 sekunde. Međutim, ovde treba dodati i vreme potrebno za skokove na novu stranicu zbog relativno malog broja redova koji se štampaju na jednoj stranici, tako da vreme koje realno treba uzeti iznosi oko 1,5 sekundi. Na taj način, za potrebe ovog izveštaja potrebno nam je oko 3.000 sekundi, ili nešto manje od jednog sata.

Sva ostala štampa, različiti pregledi nakon ažuriranja matičnih datoteka, liste grešaka, pregledi obustavljenih i neobustavljenih akontacija, izveštaji kao RAD-1 i slični, ne mogu oduzeti više od 1 sat rada štampača.

Ukupno, vreme potrebno za štampu iznosi oko 6 časova. Ako se na ovo doda i vreme potrebno za iščitavanje datoteka, koje su uglavnom sekvencijalne, vreme potrebno za čitavu obradu iznosi oko 8 časova rada sistema.

Organizacija i sadržaji datoteka

Drugi kritičan parametar pri radu sa kućnim računarima je veličina datoteka koje se obrađuju, pri čemu treba razlikovati takozvane matične datoteke od datoteka koje su radne. Pri obradi ličnih dohodaka nužno su potrebne sledeće matične datoteke:

— datoteka radnika
— datoteka OOUR-a
— datoteka kredita
— datoteka kreditora
— datoteka doprinosa
— datoteka samodoprinosa.

Analizirajmo sada sadržaj svake od ovih datoteka i procenimo njihovu veličinu.

Matična datoteka radnika

Ova datoteka treba da nam obezbedi sve potrebne podatke o radniku, kao i podatke koje ćemo koristiti pri obradi ličnih dohodaka ili nekim docnijim obradama (na primer, pri izradi obrasca M-4). Osnovni sadržaj ove datoteke mora da obuhvati:

— matični broj radnika u radnoj organizaciji (poželjno je da bude formiran kao broj OOUR-a, iza čega sledi broj iz knjige prijema radnika na rad).
— OOUR u kome radnik radi (pod uslovom da ova šifra nije sadržana u matičnom broju radnika)
— prezime i ime radnika
— lični broj radnika (iz lične karte)
— matični broj SIZ-a PIO
— datum (ili samo mesec) u kome radnik povećava staž za jednu godinu
— staž radnika
— lična kvalifikacija radnika
— potrebna kvalifikacija
— broj bodova po rešenju (ili šifra radnog mesta)
— članstvo u sindikatu
— opština i mesto stanovanja (šifre)
— prosek primanja iz prethodne godine
— primanja u poslednja tri meseca
— kumulativni iznos primanja u toku godine iz redovnog rada
— kumulativni iznos primanja u toku godine na osnovu bolovanja
— broj časova rada u toku godine
— broj časova bolovanja u toku godine
— šifra radnog mesta za priznavanje beneficiranog radnog staža
— vreme provedeno na radnom mestu sa pravom na beneficirani radni staž.

Ovako koncipiran sadržaj sloga datoteke radnika može biti smešten na 128 bajtova, što znači da svi podaci za 500 radnika mogu biti smešteni u 64Kb na disketi, ako je datoteka organizovana sekvencijalno. Sa stanovišta obrade ova datoteka može biti organizovana sekvencijalno, ali se ipak preporučuje da se organizuje kao indeksna datoteka sa ključem koji je matični broj radnika u radnoj organizaciji.

Matična datoteka OOUR-a

Broj slogova ove datoteke je jednak broju OOUR-a unutar radne organizacije. Sadržaj datoteke je:

— šifra OOUR-a
— naziv OOUR-a
— žiro račun OOUR-a.

Ukoliko radna organizacija nema OOUR-e, ova datoteka ne postoji, već se odgovarajući podaci unose direktno u program

Veličina ove datoteke je zanemarljiva, a organizacija može biti i sekvencijalna.

Matična datoteka kredita

Ova datoteka mora da nam obezbedi sve podatke o kreditu koji je radnik uzeo, kao i o svim vrstama zabrana koje postoje na njegov lični dohodak. Ova datoteka bi trebalo da sadrži sledeće podatke:

— matični broj radnika
— šifra prioriteta obustave
— šifra vrste kredita
— redni broj kredita u okviru jedne vrste kredita
— partija kredita — obustave
— iznos rate
— osnovni dug
— kumulativni iznos uplata
— datum prve rate
— period plaćanja rata
— broj perioda odlaganja.

Ovi podaci se mogu smestiti u 64 bajta, tako da računajući sa prosečno 2 obustave po radniku broj slogova iznosi 1.000, a veličina same datoteke je 64Kb. Organizacija ove datoteke može biti i sekvencijalna, ali ako se izabere indeksna organizacija, ključ datoteke je matični broj radnika.

Matična datoteka kreditora

Kao što je već rečeno, ova datoteka će sadržati oko 300 slogova sa sledećim sadržajem:

— redni broj kreditora
— naziv kreditora
— broj žiro računa kreditora.

Ovi podaci mogu se smestiti u 64 bajta, tako da je veličina čitave datoteke oko 20Kb pri sekvencijalnoj organizaciji.

Što se tiče izbora organizacije datoteke, važi isto što i pri izboru organizacije kod matične datoteke radnika, uz napomenu da su ovde izmene ipak ređe. Stoga je ažuriranje ovakve datoteke obično brže ako se radi preko sekvencijalnog upravljanja, a ne preko direktnog pristupa.

Matična datoteka doprinosa

Broj slogova koji će ova datoteka sadržati je teško unapred reći, jer zavisi od broja opština u kojima rade radnici, kao i od broja OOUR-a u radnoj organizaciji. U svakom slučaju, njihov broj neće preći 200. Sam sadržaj ove datoteke je sledeći:

— redni broj sloga (služi samo kao identifikacija za ažuriranje podataka unutar tog sloga)
— naziv doprinosa
— broj žiro računa na koji se vrši uplata
— procenat za odgovarajući doprinos
— identifikacija da li se doprinos odnosi na radnika ili pripravnika
— oznaku ko vrši plaćanje.

I ovi podaci mogu biti smešteni na 64 bajta, tako da nam je potrebno 12Kb za njeno smeštanje. Zbog svoje veličine, kao i načina korišćenja, ova datoteka može biti organizovana sekvencijalno.

Matična datoteka samodoprinosa

Broj slogova unutar ove datoteke je veoma mali, možda svega desetak, a često i manji. Sadržaj ove datoteke je:

— redni broj sloga (služi samo kao identifikacija)
— naziv samodoprinosa
— šifra opštine
— procenat za obračun
— broj žiro računa na koji se vrši uplata.

Veličina i ove datoteke je zanemarljiva, a organizacija može biti sekvencijalna.

Proces obrade

Pre nego što pristupimo objašnjenju načina obrade pretpostavimo da je matični broj radnika u radnoj organizaciji formiran tako što se na prvim mestima nalazi broj OOUR-a u kome radnik radi, iza čega sledi redni broj iz matične knjige zaposlenih. Ovo nije neko ograničenje, ali će u mnogome olakšati rad sistema (praktično, biće izbačen iz obrade izvestan broj SORT-ova), kao i naš daljnji opis obrade. Na kraju ovog dela ćemo dati i odgovarajuće izmene koje treba učiniti da bi se moglo raditi i bez ovog uslova.

Na slici 1 je prikazan dijagram obrade ličnih dohodaka, pri čemu su programi označeni sa P1, P2, P3 . . ., matične datoteke sa M1, M2, M3 . . ., radne datoteke sa R1, R2, R3 . . ., a izlazne liste sa L1, L2, L3 . . . U dijagramu nisu prikazani programi koji služe za ažuriranje matičnih datoteka, ali se

oni mogu prikazati ili kao na slici 2 (za slučaj ažuriranja indeksno organizovanih datoteka) ili kao na slici 3 (u slučaju ažuriranja sekvencijalno organizovanih datoteka). Značenje simbola M, P i L je isto kao u prethodnom slučaju. Treba napomenuti da prilikom odlaska radnika iz radne organizacije to treba evidentirati u slogu matične datoteke radnika, ali se slog ne sme ukloniti iz datoteke sve do kraja godine zbog potrebe izrade izveštaja M-4.

Ukoliko je ispunjen uslov o načinu formiranja matičnog broja radnika moguće je prilikom unosa podataka o oblicima rada u proteklom mesecu zahtevati da se to čini u rastućem redosledu matičnih brojeva. U tom slučaju program P1 omogućuje unos podataka o oblicima rada i to: broj časova rada u redovnom radu, u noćnoj smeni, prekovremeni rad u redovnoj smeni, prekovremeni rad noću, sate državnih praznika, prekovremeni rad u dane državnih praznika, plaćena odsustva, odsustvovanja sa rada na osnovu refundiranja sredstava (na primer, vojne vežbe), različite vrste bolovanja (90 ili 100 odsto), opravdane i neopravdane izostanke, paušal (ukoliko postoji), procenat kazne, iznos akontacije, iznos za korekcije (više/manje isplaćeno u prethodnom obračunu), procenat ostvarenja itd. Prilikom unosa podataka istovremeno se vrši i logička kontrola unetih podataka, tako da datoteka na izlazu (R1) sadrži samo logički ispravne podatke. Za ovu datoteku dovoljno je 128 bajtova po radniku.

Program P2 vrši uparivanje podataka iz datoteke R1 i matične datoteke radnika formirajući izlaznu datoteku R2 i listu L1 koja, radi kontrole, sadrži sve slogove koji su eventualno neupareni. Izlazna datoteka R2 pored preuzetih podataka iz datoteke R1 proširena je podacima iz matične datoteke M1 (prezime i ime radnika, broj bodova, opština i mesto stanovanja, indikatori za članstvo u SK i sindikatu, kvalifikacija, godine staža, te prosek za izračunavanje nadoknade u slučaju bolovanja), to jest sve one podatke koji će nam trebati za izračunavanje ličnog dohotka. Veličina ove datoteke će biti 128Kb, odnosno svaki slog će biti proširen na 265 bajtova koji neće biti u potpunosti iskorišćeni, ali će nam biti potrebni u toku daljeg rada.

Jedinstven program

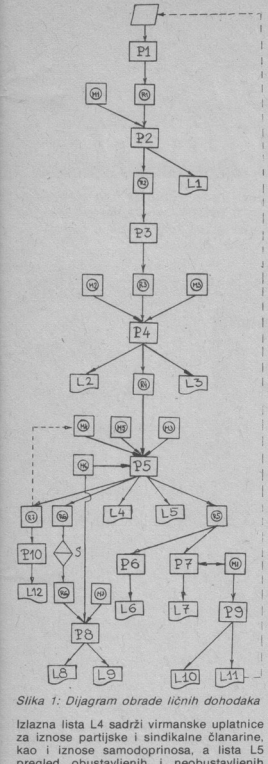
Na osnovu podataka iz datoteke P2 program P3 izračunava lični dohodak radnika u bruto iznosu, formirajući izlaznu datoteku R3. U principu je moguće zameniti programe P1, P2 i P3 jedinim jedinim programom, pod uslovom da je matična datoteka radnika organizovana indeksno i da se u toku unosa podataka o časovima rada istovremeno vrši i preuzimanje podataka iz matične datoteke radnika, te i izračunavanje vrednosti bruto iznosa ličnog dohotka.

Program P4 koristeći matične datoteke M2 (datoteke doprinosa) i M3 (datoteka OOUR-a) vrši obračun doprinosa za svaki OOUR, i na izlazu formira izlaznu listu L1 (virmani sa izračunatim doprinosima), listu L2 (rekapitulacija virmana), te izlaznu datoteku R4, koja sadrži i neto obračunati lični dohodak svakog radnika. Ukoliko se želi prikaz svakog pojedinačnog doprinosa na koverti radnika potrebno je na ovom mestu to i uraditi. Međutim, to bi znatno povećalo broj bajtova za radnu datoteku te se to praktično teško može ostvariti na disketama kapaciteta reda stotinak kilobajtova (ali svakako, bez većih problema sa vinčester diskovima). Napomenimo da je moguće i ovaj program uključiti u prva tri programa, u jedinstven program. Međutim na 8-bitnim računarima to je pre naučna fantastika nego realnost, ali na 16-bitnim mašinama lako ostvarljivo. Međutim, veoma je realno očekivati od veštog programera da prva dva programa formira kao jedan program, a da programe P3 i P4 sastavi takođe u jedan jedinstven program.

Program P5 služi za obračune svih obustava na ličnom dohotku radnika (sudsko-administrativne zabrane, kredite, sindikalnu i partijsku članarinu, akontacije i slično). Za te potrebe koriste se, pored ulazne datoteke R4 i matične datoteke M4 (matična datoteka kredita), M5 (matična datoteka samodoprinosa), M3 (matična datoteka OOUR-a) i M6 (matična datoteka kreditora). Kako je matična datoteka kredita sortirana po matičnom broju radnika, to je njenim uparivanjem sa datotekom R4 moguće na izlazu dobiti radnu datoteku R5, koja bi sadržala iznose obustavljenih kredita, a ako raspolažemo sa dovoljno prostora na disketi (da povećamo slog na 384 bajta, odnosno čitavu datoteku R5 na 192Kb) i sa nazivom kreditora i saldom kredita. Druga izlazna datoteka R6 sadrži sledeće podatke: broj kreditora, prezime i ime radnika, partiju kredita, iznos obustave, saldo kredita, kao i još neke značajne podatke koji se žele saopštiti kreditoru. Slog ove datoteke ne prelazi 64 bajta, tako da možemo računati da njena veličina ne prelazi 64Kb (računajući u proseku sa 2 kredita po jednom radniku).

Godišnja obrada

Izlazna datoteka R7 predstavlja u stvari ažuriranu matičnu datoteku kredita, i u obradi za sledeći mesec će biti na mestu ulazne datoteke M4. Veličina ove datoteke je ista kao i datoteke M4, to jest 64Kb.



Slika 1: Dijagram obrade ličnih dohodaka

Izlazna lista L4 sadrži virmanske uplatnice za iznose partijske i sindikalne članarine, kao i iznose samodoprinosa, a lista L5 pregled obustavljenih i neobustavljenih akontacija radnika. Sa stanovišta angažovanog kapaciteta na disketama ovaj program postavlja najveće zahteve. Naime ukupna veličina datoteka je oko 440Kb [64Kb (M4)+1Kb (M5)+1Kb (M3)+20Kb (M6)+192Kb (R5)+64Kb (R6)+64Kb (R7)], plus nekoliko kilobajtova za privremeni smeštaj datoteke koja sadrži štampu izveštaja L5. Ukoliko je datoteka M4 (matična datoteka kredita) organizovana kao indeksna, to se gornji zbir može smanjiti za 64Kb, jer bi otpala potreba za formiranjem datoteke R7, a datoteka M4 bi bila ulazno izlazna datoteka u program P5. Ukoliko se ne raspolaže ni sa ovolikim prostorom, program P5 bi morao biti razdvojen na dva nezavisna programa.

63/plavi koverti iz računara

Program P6 vrši jednostavno izlistavanje ulazne datoteke R5, na unapred pripremljeni koverat (lista L6). Program P7 vrši ažuriranje podataka u matičnoj datoteci radnika, menjajući sadržaje polja o kumulativnom iznosu ličnog dohotka radnika u tekućoj godini (kako za rad tako i za bolovanja), kumulativnom iznosu broja časova rada i bolovanja, te vršeći pomeranje ažurira primanja u prethodna tri meseca. Kao izveštaj (L7) javlja se potreban statistički pregled ličnih dohodaka (na primer, izveštaj RAD-1).

Pre ulska u program P8 potrebno je datoteku R6 sortirati po šifri kreditora. Tako pripremljena datoteka, uz korišćenje matične datoteke kreditora (M6) i matične datoteke OOUR-a (M3) služi za izradu virmanskih uplatnica koje se upućuju kreditorima (L8) zajedno sa izveštajem o izvršenim uplatama (L9).

Sa ovim bi obrada ličnih dohodaka za jedan mesec bila završena. Međutim, potrebna su nam još dva programa za pripremu obrade u sledećem mesecu. Program P9 vrši izlistavanje matične datoteke radnika (po potrebi — L10), i obavezno izveštaj L11, koji u stvari predstavlja osnovu za unošenje podataka o radu radnika u sledećem mesecu, to jest dokument sa koga će se nakon unošenja časova rada vršiti obuhvat podatka za program P1. Program P10 izlistava opciono ili sve kredite radnika, ili kredite koji su isplaćeni (saldo nula), kako bi se oni uklonili iz matične datoteke kredita (izveštaj L12).

Konačno, treba reći da je za godišnju obradu potrebno predvideti još jedan program koji bi radio sa matičnom datotekom radnika. Cilj ovog programa je da izradi obrazac M-4, te da izvrši potrebna ažuriranja u matičnoj datoteci: anuliranje odgovarajućih polja kumulativa primanja i časova rada, te izmenu primanja u toku prethodne godine. U toku ove obrade vrši se i izbacivanje slogova radnika koji su napustili radnu organizaciju u toku prethodne godine.

Manje vreme obrade

U prethodnom delu dat je jedan od mogućih načina organizovanja obrade ličnih dohodaka. Predloženo rešenje, bez ikakvih pretenzija da je najbolje moguće, ima nekoliko pogodnosti. Kao prvo, broj SORT-ova je smanjen praktično na minimum, što ima za posledicu znatno smanjenje vremena obrade. Mnogo je važnija, međutim, činjenica da pri radu bez SORT-ova imamo mogućnost rada sa datotekama teorijski neograničene dužine. Ovo bi imalo svoj puni značaj kada bismo raspolagali sa tri disketne jedinice, pri čemu bi dve služile za ulaznu i izlaznu datoteku, treća jedinica služila za matične datoteke. Međutim, naša konfiguracija to ne dozvoljava, jer smo na početku odredili da ćemo raspolagati samo sa dve disketne jedinice.

Ipak, analizirajmo ovu prednost posmatrajući dijagram obrade sa slike 1. Svi naši programi, osim P2, P4, P5 i P8 imaju po jednu datoteku na ulazu i izlazu, tako da njihova dužina u odnosu na kapacitet disketne jedinice praktično nema nikakvog uticaja. Kod programa P2, gde u obradi učestvuje i matična datoteka radnika M1, problema ne bi trebalo da bude jer dužina obe ulazne datoteke može biti komotno smeštena na jednoj disketi, dok bi druga disketna jedinica služila za izlaznu datoteku. U programu P4, kod koga učestvuju dve matične datoteke (M2 — datoteka doprinosa i M3 — datoteka OOUR-a) možemo na početku programa učitati direktno u memoriju podatke o vrsti doprinosa i njegovom procentu, iz datoteke M2, datoteku zatvoriti i na njeno mesto postaviti datoteku R3. U toku čitanja te datoteke, i izračunavanja neto ličnog dohotka radnika možemo formirati kumulative za iznose pojedinih doprinosa u okviru tabele unutar memorije.

Nakon iščitavanja datoteke R3 i formiranja datoteke R4 u disketne jedinice se ponovo stavlja datoteka M2 i M3. Sekvencijalnim pražnjenjem formirane tabele i istim takvim čitanjem datoteka M2 i M3 dobijaju se izveštaji L2 i L3. Problem je nešto složeniji kod programa P5, jer u obradi učestvuje čak 7 datoteka. Međutim, već smo rekli da je moguće ukloniti datoteku R7 ako se datoteka M4 (matična datoteka kreditora) organizuje kao indeksna. Datoteka M3 (datoteka OOUR-a) i M5 (datoteka samodoprinosa) veoma su male i moguće ih je na početku programa učitati u memoriju kompletne, tako da njihovo prisustvo na disketama nije nužno potrebno u toku obrade. Na taj način se broj stalno prisutnih datoteka smanjuje na četiri, što je još uvek mnogo. Ukoliko se, pak, odustane od zahteva da se na koverti prikazuje naziv kreditora, otpada i potreba za datotekom M6 (matična datoteka kreditora).

Programski paketi

Ako se na tome ipak insistira, moguće je ujediniti matičnu datoteku kredita M4 sa matičnom datotekom kreditora pre ulaska u program P5, što bi imalo za posledicu proširenje podataka u datoteci M4 za naziv kreditora i njegov broj žiro računa. Pa i u tom slučaju nam ostaju tri datoteke za obradu. Verovatno je jedino rešenje da se na izlaznoj disketi nade onoliko puta datoteka M4 koliko nam je disketa potrebno za smeštaj čitave izlazne datoteke R5. Rešenje nije baš lepo, ali nužda zakon menja. Što se tiče programa P8, problema nema, jer se datoteka M3 (datoteka OOUR-a) može učitati u memoriju na samom početku programa i odatle se mogu koristiti podaci koji su nam potrebni. Konačno, treća prednost predloženog načina obrade je u relativno malom broju programa, zbog čega je njihovo održavanje u eksploataciji veoma olakšano.

Ukoliko se iz bilo kog razloga ne može prihvatiti uslov o načinu organizovanja matičnog broja radnika, dijagram obrade sa slike 1 mogao bi da pretrpi izvesne promene, uzrokovane potrebom za uvođenjem nekoliko SORT-ova. Datoteku R1 bi trebalo sortirati po matičnom broju radnika, datoteku R3 po broju OOUR-a, a datoteku R4 ponovno po matičnom broju radnika. Uvođenje SORT-a ne samo što bi produžilo vreme obrade, već bi i onemogućilo osnovnu prednost navedenog rešenja da radne datoteke mogu biti neograničene dužine.

Konačno, recimo još jednom, cilj ovog članka nije bio da pruži gotovo rešenje za obradu ličnih dohodaka, već samo da ukaže na potencijalne mogućnosti primene kućnih računara u jednoj oblasti gde danas neprikosnoveno vladaju mnogo moćniji sistemi. Autoru je poznato da se danas nude programski paketi za obradu ličnih dohodaka na kućnim računarima, ali nije imao priliku da se ni sa jednim od rešenja bliže upozna. Ovaj članak zato predstavlja i poziv svim zainteresovanima da preko „Računara" upoznaju širi auditorijum sa svojim gotovim rešenjima.

Mihailo Karapandžić, dipl. inž.

ima li programera u avionu

nastavak sa str 14

• Iz vašeg odgovora bi se moglo zaključiti da za pravljenje domaćeg računara do sada nisu angažovani pravi ljudi. Ako bi neko ponudio Elektrotehničkom fakultetu da napravi računar, da li bi kadrovi koji su trenutno tu bili u stanju da u nekoj klasi naprave računar bolji od onih kojima trenutno raspolažemo?

— Siguran sam da bismo mogli.

• A za koje vreme i po kojoj ceni?

— Pa, mislim, relativno brzo. Ne zaboravite da bismo moje kolege i ja u takav projekat uložili desetine godina minulog naučnog rada i iskustva. A što se cene tiče, reći ću vam šta je Mocart rekao kada su ga upitali koliko treba da mu plate za jednu njegovu kompoziciju. „Ma koliko mi platili, precenićete sadašnji trud i potcenićete moj talenat". Ja mislim da treba dobro platiti kvalitetno delo, ali o vrednovanju naučnog rada bolje da ne govorimo.

• U čemu je naša prava šansa kada su računari u pitanju?

— Izlaz iz ekonomske situacije bez primene računarstva nije moguć. Recimo da neka radna organizacija treba u roku da odgovori na neki tender iz Afrike ili sa Bliskog istoka. Treba graditi, na primer, hiljadu transformatora za određene meteorološke prilike. Za to je neophodno imati već gotove programe za projektovanje, kretanje cena bakra i gvozdenog lima na londonskoj berzi i niza drugih paramtetara. Rezultati optimizacije treba hitno da pruže odgovor na pitanje pod kojim uslovima ući u taj posao. Ako je posao potcenjen radiće se u korist sopstvene štete i „Feni"-čani će imati nove sledbenike, a ako je precenjen — inostrani ponuđač će dati bolje uslove i preuzeti posao. Pre ove analize očigledno treba otkloniti greške u uvezenom softveru. Sovjeti su prvi poleteli u kosmos zahvaljujući tehnologiji Zapada i sopstvenoj pameti. Ni mi u pameti nikada nismo oskudevali.

• Kada smo već kod pameti, osvrnimo se i na glavnu primenu računara pete generacije, područje tzv. „veštačke inteligencije". Upravo je „veštačka inteligencija" najviše podstakla maštu pisaca i reditelja naučne fantastike, pred čijim delima mnogi ljudi osećaju strah od „inteligentnih računara" koji su u stanju da samostalno preuzmu kontrolu nad životima običnih smrtnika. Šta vi kao profesionalac kažete, kada će računari moći da misle i samostalno donose odluke?

— Još dosta davno načinio sam program koji je mogao sam sebe da koriguje i u svakom novom prolazu brže daje izlazne rezultate. Za njega bi neko neupućen rekao da spada u domen „veštačke inteligencije". Ali ono što moramo da naglasimo kada se govori o ovom programu i „veštačkoj inteligenciji" jeste da računar čini upravo ono što su njegovi programeri predvideli. A na pitanje kada će računari moći da misle i samostalno donose odluke kao čovek, moj odgovor je — onda kada avioni (koji poput ptica lete) budu počeli da svijaju gnezda i nose jaja kao ptice.

Najviše smo govorili o programerima. Šta biste na kraju rekli o profesiji programera?

— Isto što i o pilotima. Nijedan pilot nije napustio letenje po svojoj volji. Nije mi poznato da je neki programer prestao da programira. To je posao koji zaokupljа celu ličnost i pruža veliko intelektualno zadovoljstvo.

* * *

Zahvaljujemo profesoru Slaviću na razgovoru i nadamo se da neće zaboraviti obećanje o saradnji koje je dao za naše čitaoce. Posebno cenimo spremnost da našim čitaocima prenese saznanja do kojih je došao višegodišnjim upornim radom. Očekujemo uskoro njegove priloge, ali i vaša pitanja vezana za računanja na računaru.

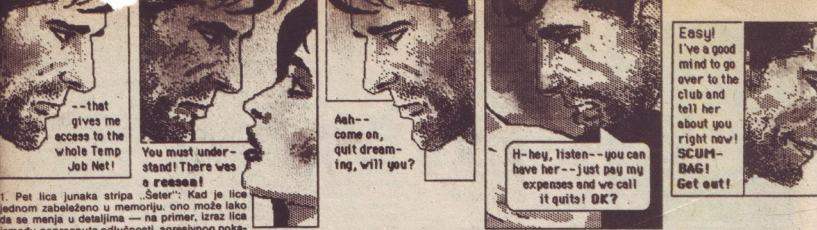
Razgovor vodila: Nevenka Spalević

Računari i umetnost

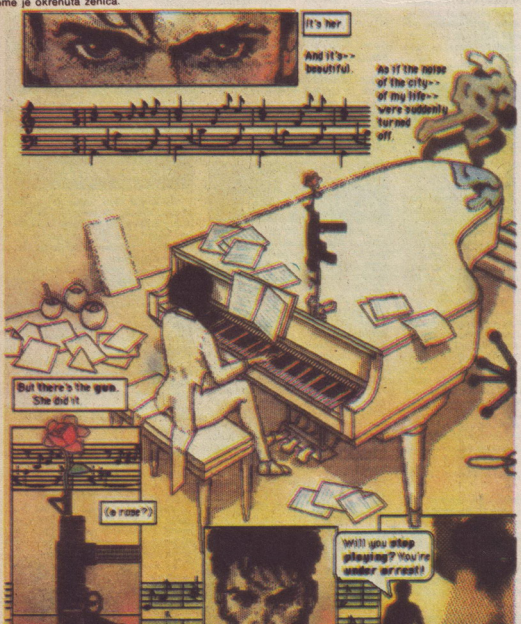
stripoteka na kompjuteru

Prodirući u sve pore čovekovog života i rada, računar nije mimoišao čak ni područje u kome su tradicionalno vladali olovka, pero i mastilo — područje stripova. Računar je napravio tek prvi korak u svet devete umetnosti, ali korak koji mnogo obećava.

2. Jedna kompletna tabla iz stripa „Šeter": Čovek sa crnim konjskim repićem i zapaljenom cigaretom pojavljuje se tri puta. Slika u sredini pokazuje koliko je lako neki lik „zumirati" — načiniti ga po volji većim ili manjim. Ubacivanje teksta obavlja se istovremeno.



1. Pet lica junaka stripa „Šeter": Kad je lice jednom zabeleženo u memoriju, ono može lako da se menja u detaljima — na primer, izraz lica između napregnute odlučnosti, agresivnog pokazivanja zuba i nagoveštenog osmeha; isto tako, lako je promeniti svetlost i senke, ili pravac u kome je okrenuta zenica.



3. Veliki crtež sa umetnutim manjim slikama i crtežima. Ako se crtaču ne dopada poredak pojedinih slikovnih elemenata, on ih može mnogo lakše izmeniti nego prilikom crtanja na papiru. Sa računarom „mekintoš" jedna stvar još uvek ne može da se ostvari: kolorisanje; boje moraju da se nanose rukom.

„Sada ću sigurno bankrotirati", rekao je Rik (Rick) Oliver, glavni urednik firme „Prvi stripovi" u gradu Evanstonu, SAD. Pred njim je ležao Šeter" (.Shatter") — prvi strip načinjen na ličnom računaru. „Nikada nisam mogao poverovati da bi neko pomoću „meka" mogao ovako nešto ostvariti".

Pod „mekom" Oliver je, dabome, podrazumevao računar „mekintoš", A Majkl Saenz (Michael Saenz), crtač „Šetera", proizveo je čitavu priču u slici na svome „meku". Sem svog talenta i svojih ideja. Saenz je morao za tu svrhu da poseduje i dobro poznavanje kompjutera i specijalnog grafičkog softvera „Macpaint".

Na 28 strana Saenz opisuje avanturu „Šetera", policajca iz potpuno kompjuterizovanog 21. veka, koji u svojim letećim patrolnim kolima kreće u potragu za raznim otpadnicima od zakona, čije pištolje bombe sa ugrađenim mozgom pacova ugorčavaju red i zakon.

Jednako dovitljiv kao Šeter, morao je sa svojom grafičkom kompjuterskom tablom da bude i njegov tvorac Saenz. Jer crtač iz Čikaga je sa tim prvim profesionalno načinjenim stripom na računaru izazvao pravu revoluciju u svojoj branši. Više od četrdeset godina crtači stripova koristili su isto oruđe: olovku i pero, mastilo i tehniku prskanja. „A pre tehnike prskanja, prethodnici crtača stripova, politički karikaturisti, radili su pomoću litografija i duboreza", kaže Majkl Saenz. „Od pronalaska tehnike prskanja u proizvodnji stripova ništa novo nije učinjeno".

Sada stvari stoje drukčije. Sa računarom crtač može mnogo racionalnije da radi. Jer upravo kod stripova često se ponavljaju određene slike i motivi, tim što se ne retko samo malčice menjaju. Uzmimo kao primer lice glavnog junaka. Da li se on smeška ili ceri — to uglavnom pokazuje nekoliko malih izmena na izrazu njegovog lica. Te nijanse morale su do sada da se uvek iznova crtaju, bez obzira na to što je glavni deo posla otpadao na šablone.

Drukčije sve to izgleda kod Majkla Saenza. On može svaku sliku da zabeleži u računaru i da je pritiskom na tastaturu po svojoj volji izmeni, da se pojedinim komponentama „poigra" i po svome nahođenju ih poveća, umanji ili načini isečke — a da pri svemu tome ne izgubi nijednu sliku ili da je umrlja mastilom; jer originali nisu „memorisani" na papiru, nego u elektronskoj memoriji računara. Crtač stvara pomoću kompjutera crno-belu sliku, koja je već spremna za kolorisanje; ova skica za dobijanje čistog crteža pri tome postaje nepotrebna. Istovremeno, Saenz ne mora da se trudi oko stvaranja specijalnih efekata, jer je ta veština sadržana u računaru i samo čeka na to da Saenz dâ punog maha svojoj kreativnosti.

Istini za volju, treba priznati da „Šeter" deluje pomalo „tvrdo". Kad se prikazuje unutrašnjost sobe, zgrada ili vozilo, čovek pre ima utisak da posmatra kompjutersku igru na ekranu. A mnoga zasenčenja na slikama sastoje se iz jasno raspoznatljivih tačkica — kao da su stotine muva stavile svoj šapice na papir.

ali čini se da ljubitelji stripa ne vide u ' neki nedostatak. U međuvremenu je prvo iz „Šetera" rasprodato do komada u tira' 60.000 primeraka. Sada je u toku štar drugog izdanja ovog prvog kompju stripa.

Priprema
Branko Đaković

Anđelka u knjigu!

Ponovo je članak Anđelka Zgoreleca bio jedan od najboljih tekstova u „Računarima“. Već duže vreme pratim tekstove tog čoveka i ne mogu da se otmem utisku da njegova javljanja iz Londona dobijaju nezasluženo malo prostora u odnosu na njihov kvalitet. No, ja vas verovatno ne bih gnjavio time da mi nije još nešto palo na pamet u vezi sa Anđelkom Zgorelecom.

Ideju sam dobio kada sam video umetak sa „spektrumovim“ rutinama Jovana Skuljana. Pomislio sam da bi to bilo dobro videti kao knjigu. Nisam previše o tome razmišljao, ali mi je odmah pala na um slična ideja i u vezi sa Anđelkom Zgorelecom. Zar ne biste mogli da izdate knjigu Anđelkovih tekstova, koja bi predstavljala pravo malo osveženje u inače smrtonosnoj literaturi s područja računara. Smatram da eseji Anđelka Zgoreleca (a to jesu pravi eseji) imaju daleko više opravdanja da se pojave kao knjiga nego bilo kojekakve dosta glupave računarske knjige koje sam nedavno video u knjižari (da ne spominjem naslove i imena). S druge strane, verovatno bi se i vama isplatilo, jer, siguran sam, interesovanje bi bilo veliko.

Toliko od mene, a vi razmislite.

Goran Aleksić, 18000 Niš

Prelazak „Računara“ na mesečno izlaženje otvara nove mogućnosti, svakako veće nego pre toga, i na području izdavanja raznih brošura, priručnika i knjiga manjeg obima, o čemu nešto opširnije pismo na str. 25. Što se ideje o knjizi tekstova Anđelka Zgoreleca tiče, imaćemo je u vidu.

Rutine koje život znače

To da su „Računari“ poskupeli primetio sam odmah na kiosku. Nije bilo naročito teško. Prestalo je da mi smeta čim sam primetio šta se nalazi u „Računarima“. Vaš umetak, ako smem tako da se izrazim, „Sve spektrumove rutine“, zaista je prava stvar i neverovatan pogodak. Priznajem da bih i samo za njega platio 250 dinara, ali nemojte da to sad iskoristite pa da dignete cenu još više. Već dugo očekujem da se pojavi nešto kao ovaj umetak. Smatram da bi bilo izuzetno dobro, a za vas veoma isplativo, da u svakom broju imate dodatak (to jest umetak) koji bi po kvalitetu i interesantnosti odgovarao ovom iz „sedmice“.

Još nešto. Posle izvanredne naslovne strane u „šestici“ ova u „sedmici“ je zaista izuzetno ružna. Da li vi to radite namerno?

Željko Tonšić, Prvomajska 16, 11080 Zemun

„Rutine“ su doista naporno i ambiciozno pripremane. Drago nam je da ti se dopadaju. Nadamo se da su i ostali kojima su se „Rutine“ svidele lako preboleli poskupljenje na koje smo zaista bili primorani (jeste fraza, ali je barem istinita). O naslovnoj strani bilo je i lepih mišljenja.

Igre kao ugrožena vrsta

Da vi mrzite video igre to i nije tako teško zaključiti. U svakom novom broju sve je manje pregleda novih video igara. Nisam baš zaluđen za video igre, kako bi se iz mog pisma mogło zaključiti, ali smatram da one predstavljaju izvanredno razonodu i zabavu, a osim toga, računari su svoj prodor kod nas postigli upravo zahvaljujući video igrama.

Možda ćete se braniti da prostor čuvate za važnije teme, ali kad se pažljivo pregledaju „Računari“ vidi se da svakakve čudnovate teme imaju mesta, a samo se igrama iz broja u broj smanjuje prostor. Od nekadašnjih dve strane prešlo se na jednu i po, a sada i na samo jednu. Ako tako nastavite, uskoro ih uopšte neće ni biti u „Računarima“. Ako imate problema sa nabavljanjem i ocenjivanjem najnovijih igara obratite se pravim ljudima — piratima. Samo napišite šta vam treba i ja sam siguran da ćete mesečno dobiti po nekoliko stotina igara. Eto, to vam je recept, a ostaje samo još da vi to objavite.

Milojko Blagojević. Cerova. 31230 Arilje

Ima, dragi Milojko, u redakciji ljudi koji vole igre isto toliko koliko i ti. Činjenica da je u „Računarima“ sve manje igara posledica je toga što pokušavamo da se, osim kvalitetom tema u celini, i time distanciramo od pojedinih časopisa koji žive praktično od video igara. Igara će biti i dalje, nekad više, nekad manje.

Dejan iz niotkuda

„Računare“ čitam od prvog broja. To ne kažem tek tako, nego želim da vidite da sam pažljivo pročitao sve brojeva pre nego što sam došao do sledećeg zaključka: osoba koja je u „Računarima“ potpisana kao Dejan Ristanović — ne postoji! Verovatno se radi o nekoj vrsti redakcijskog pseudonima koji koriste vlastiti urednici i saradnici. Ne znam samo zašto to radite? Zašto nam servirate nekog fiktivnog Dejana Ristanovića? Možda se radi o skromnosti ili o honorarima ili o nečem trećem, ali jedno je sigurno: da ste malo bolje pazili, ne bi vas ni sada otkrili.

Moj brat i ja skupljamo u fascikle sve interesantne materijale iz „Računara“. Pre par nedelja smo zajedno ustanovili da najviše materijala ima u fascikli sa imenom Dejana Ristanovića. Zainteresovani izvadili smo sadržaj i pregledali ga. Ispostavilo se da je u poslednjih godinu i nešto dana „Dejan Ristanović“ napisao i objavio materijala koliko i svi ostali saradnici „Računara“ zajedno. Po tome ispada da on niti spava „niti jede niti pije niti upravo zahvaljujući njegovim tekstovima „Računari“ i „Galaksija“. Što je i od „Računara“, mnogo je. Čestitajte nam otkriću i priznajte zašto ste to radili.

Goran (i Marko) Stevanović, 34000 Kragujevac

Dejan, razume se, postoji. Činjenica da on napiše materijala koliko i svi ostali saradnici zajedno pre je argumentan pokazatelj — neprijatelja . . . Ipak, bio je to dobar pokušaj, a i znak da pažljivo čitate „Računare“. Samo tako nastavite.

Jelena — poslednji put

Iskreno verujem da Jelena zna više od pisanja osim testova i jako bih se obradovao ako bi u sledećem broju naišao na nešto što po kvalitetu podseća na radove Jovana Skuljana, Dejana Ristanovića i Vladimira Kostića.

Kao drugo, verovatno ste primetili da po izboru omiljenih pisaca tekstova u vašoj reviji spadam u vlasnike „spektruma“, ali imao sam prilike da radim i na „šnajderu CPC 464“ i mašina me je oduševila. Bejzik je odličan i vrlo prijatan za rad, a mašina deluje profesionalnije i neuporedivo ljubaznije od „komodora 64“. Zato vas molim da napišete nešto više o toj mašini i njenoj memorijskoj mapi, zatim pričaju delu memorije koji se preklapa i uopšte o njegovom operativnom sistemu. Objavite adresu Dragana Gribića, a ja ću čak i doplatiti pri trampi „spektruma 48K“ za „amstrada CPC 464“ kako bih ga oslobodio „bede“.

Đorđe Radovanović. Narodnog Fronta 34. 21000 Novi Sad

Većina naših nestrpljivih čitalaca nije pažljivo pročitala ime ove rubrike. Oni pažljivi primetili su da se ona zove* Load „pismo“, *a ne* Load „Jelena“ *kao što to neki izgleda, još uvek misle. Da bi smo takve i slične razuverili i uputili na neku novu žrtvu — kao što je Sonja Savić ili Suzana Mančić — odlučili smo da u sledećih stotinak brojeva ne objavljujemo pisma koja se tiču J. R.

Vlada postao humanista

Čitajući poslednjih nekoliko brojeva, sa zaprepašćenjem sam primetio da vi otupljujete svoju softversku oštricu. Pored nekoliko dužih sarađnika koji su do sad pisali tekstove nekako humanistički orijentisane (ja to tako vidim, ali nemojte da zamerite ako sam pogrešio termin), sada je i Vlada Kostić, vaša prva softverska-hakerska perjanica, prešaltavan na neki tekst gde je primoran da piše stvari koje verovatno ne bi želeo. Zašto ga mučite? Zar nije bolje da ne osiromašujete vaš fond softveraša, a da teoriju i praksu prepustite onima koji nisu tako jaki u programiranju. Sećate li se vas. Uskoro će Dejan Ristanović i Vlada Kostić pisati socioloških analize računara, a Jelena Rupnik, Branko Đaković i Duško Savić programe koji aman baš ništa neće valjati.

Zato nemojte da namećete ljudima ono što ne žele — biće bolje i po vaš časopis.

Franjo Šetlef. Beogradski put 138, 41000 Zagreb

Dragi Franjo, oprosti, nećemo nikad više. Vlada ima da piše programe i da zaboravi humanističke nauke.

Delikates za iščitavanje

Vi zaista uvek imate po nekog keca u rukavu. U „sedmici“ taj kec je bio jedan pravi delikates koji sam do sada pročitao bar pet puta. Naravno, to je onaj intervju sa Stivenom Džobsom.

Iako sam uglavnom zainteresovan za hardver, a nešto manje za softver, a za pričalački deo „Računara“ ne marim preterano, ovaj intervju mi je drazi od svih hardverskih pregleda u „sedmici“, a čao bih za njega i tri „biblioteke programa“.

E sad, imao bih i ja neki predlog, koji se ukratko svodi na to da u svakom broju imate bar po jedan ovakav intervju. Znam da nije lako naći čoveka kao što je Stiven Džobs, ali ostaje još puno zanimljivih ljudi u računarskoj industriji koji imaju šta da kažu (Sinkler, Tramiel ...). Povoljna činjenica za vas je to što izlazite tek svaka dva meseca, pa imate dovoljno vremena da nabavite zaista super intervju kakav je ovaj bio.

Ako vas mrzi da putujete po svetu i intervjuišete ljude, pošaljite mene. Ili raspišite javni konkurs za najbolji računarski intervju. Ili malo bolje prelistajte strane časopise. u njima mora da ima gomila sličnih intervjua.

Žarko Đukić, Mornarička 2, 78280 Glamoč

Kada je o „pogodnostima“, tvoj sud je zapravo usamijen; dragi čitaoci nam uši probiše žalopojkama o nedostacima dvomesečnog izlaženja. Od ovog broja pa nadalje biće po njihovom: „Računari“ su postali mesečnik. Inače, nama nije dovoljno da obezbedimo običan intervju sa računardžijom. Više volimo da on bude takav da i hardver-softveraš kakav si ti oseti da je intervju zaista super. A do takvih sagovornika teže se dolazi.

1

2

3

4

5

6

NARUDŽBENICA

Galaksija, Bulevar vojvode Mišića 17, 11000 Beograd

Ovim neopozivo naručujem sledeća specijalna izdanja časopisa „Galaksija": 3—4—5—6 (Računari) — 8 (Tesla), po ceni od 200 dinara po primerku, odnosno 7 (Računari), po ceni od 250 dinara (zaokružiti odgovarajuće brojeve). Iznos od ukupno _____ dinara uplatiću poštaru prilikom preuzimanja pošiljke — POUZEĆEM.

Ime i prezime _____

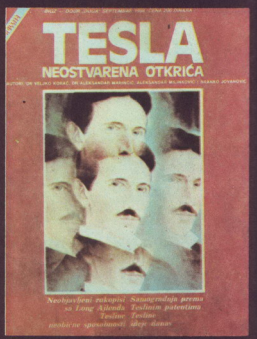
Ulica i broj _____

Broj pošte i mesto _____

Datum

Potpis

7



Mercator

MEDNARODNA TRGOVINA
TOZD CONTAL
61000 LJUBLJANA, Titova 66

CONTAL **ZASTUPA I PRODAJE** **SHARP**

Iz programa ovog uglednog proizvođača, predstavljamo vam:

PERSONAL COMPUTER MZ — 731

Široka primena: od hobija do posla i nauke.

— 64 K RAM
— priključenje na TV ili jedan od SHARP-ovih display modela.
— Vrhunska tastatura koja omogućava lako rukovanje

CENA: DM 1.100.-
Cena opreme: Display 420.- i 840.-DM
Priručnik DM 28

Informacije na tel: (061) 328-441; 318-311